

## GRAPHIC ELEMENT WITH MULTIPLE VISUALIZATIONS IN A PROCESS ENVIRONMENT

Publication number: CN101893861 (A)

Publication date: 2010-11-24

Inventor(s):

Applicant(s):

Classification:

- international: **G05B15/02; G05B19/042; G05B23/02; G06F9/44; H04L29/08**


- European: G05B19/042P; G05B19/418S; G05B23/02; G06F9/44G4; G06F9/44M; G06N5/04; G06Q10/00C; G06Q50/00B; H04L12/24E; H04L12/24F1; H04L12/24F3; H04L29/08N11; H04L29/08N15; H04L29/08N27; H04L29/08N27X3; H04L29/08N35


Application number: CN20101254958 20050504


Priority number(s): US20040567980P 20040504


Also published as:

 WO2005109122 (A1)

 WO2005109131 (A1)

 WO2005109130 (A1)

 WO2005109129 (A1)

 WO2005107416 (A2)

more >>

Abstract not available for CN 101893861 (A)

Abstract of corresponding document: WO 2005109122 (A1)

Smart graphic elements are provided for use as portions or components of one or more graphic displays, which may be executed in a process plant to display information to users about the process plant environment, such as the current state of devices within the process plant. Each of the graphic elements is an executable object that includes a property or a variable that may be bound to an associated process entity, like a field device, and that includes multiple visualizations, each of which may be used to graphically depict the associated process entity on a user interface when the graphic element is executed as part of the graphic display. Any of the graphic element visualizations may be used in any particular graphic display and the same graphic display may use different ones of the visualizations at different times. The different visualizations associated with a graphic element make the graphic element more versatile, at they allow the same graphic element to be used in different displays using different graphical styles or norms. These visualizations also enable the same graphic element to be used in displays designed for different types of display devices, such as display devices having large display screens, standard computer screens and very small display screens, such as PDA and telephone display screens.

.....  
Data supplied from the **espacenet** database — Worldwide



1. 一种在过程环境内使用可执行图形显示的方法,包括:

创建一个或多个可执行图形显示,使其包括将要绑定到加工厂内数据源的一个或多个可视描述以及一个或多个参数;

为所述一个或多个可执行图形显示中的每一个存储一角色;和

根据为所述一个或多个可执行图形显示所定义的角色,将所述一个或多个可执行图形显示分配给显示设备。

2. 根据权利要求1的方法,进一步包括,根据所述一个或多个可执行图形显示被分配给所述显示设备的方式,下载所述一个或多个可执行图形显示。

3. 根据权利要求1的方法,其中,为所述可执行图形显示中的每一个存储一角色的步骤包括:为所述可执行图形显示中的一个存储一操作界面角色。

4. 根据权利要求1的方法,其中,为所述可执行图形显示中的每一个存储一角色的步骤包括:为所述可执行图形显示中的一个存储一维护界面角色。

5. 根据权利要求1的方法,其中,为所述可执行图形显示中的每一个存储一角色的步骤包括:为所述可执行图形显示中的一个存储一仿真界面角色。

6. 根据权利要求1的方法,其中,将所述一个或多个可执行图形显示分配给显示设备的步骤包括:根据为所述一个或多个可执行图形显示存储的角色,将所述一个或多个可执行图形显示分配给所述显示设备的一个或多个功能子系统。

7. 根据权利要求1的方法,其中,将所述一个或多个可执行图形显示分配给显示设备的步骤包括:根据与显示设备关联的设备,将所述一个或多个可执行图形显示分配给该显示设备。

## 过程配置和控制环境中的图形集成

[0001] 相关申请

[0002] 本申请是 2004 年 5 月 4 日递交的顺序号为 60/567,980、题为“用于表示、监测和与过程控制系统交互的图形用户界面”的美国临时专利申请的正式递交申请，并出于优先权的目的要求其权益，在此本申请明确地将其全部内容合并作为参考。本申请还与 2003 年 7 月 21 日递交的、题为“加工工厂中图形显示元素、过程模块和控制模块的集成”、并在 2004 年 8 月 5 日以美国出版号 2004/0153804 出版的、顺序号为 10/625,481 的美国专利申请有关，该申请为 2002 年 10 月 22 日递交的、在 2004 年 4 月 22 日以美国出版号 2004/0075689 出版的、题为“加工工厂中的智能过程模块和对象”的、顺序号为 10/278,469 的美国专利申请的部分继续申请，在此明确地将这两个公开的全部内容合并作为参考。本申请还与 2003 年 2 月 18 日递交的、题为“加工工厂配置系统中的模块类对象”、并在 2004 年 10 月 7 日以美国出版号 2004/0199925 出版的、顺序号为 10/368,151 的美国专利申请有关，在此明确地将其全部内容合并作为参考。本申请还和与本申请在同一天递交的正在作为国际 (PCT) 申请的下列专利申请有关，在此明确地将这些专利申请全部内容合并作为参考：“过程环境中的关联图形显示”(代理备案号 06005/41111)；“用于过程控制系统的用户可配置警报和警报趋势”(代理备案号 06005/41112)；“加工工厂中过程模块和专家系统的集成”(代理备案号 No. 06005/41113)；“集成环境中具有定制的过程图形显示层的加工工厂用户界面系统”(代理备案号 No. 06005/41114)；“过程环境中的脚本图形”(代理备案号 No. 06005/41115)；“过程环境中具有多种显像的图形元素”(代理备案号 No. 06005/41117)；“加工工厂中用于配置图形显示元素和过程模块的系统”(代理备案号 No. 06005/41118)；“用于统一的过程控制系统界面的图形显示配置框架”(代理备案号 No. 06005/41124)；“加工工厂用户界面中基于 Markup 语言的动态过程图形”(代理备案号 No. 06005/41127)；“用于修改过程控制数据的方法和装置”(代理备案号 No. 06005/591622 和 No. 20040/59-11622)；“用于访问过程控制数据的方法和装置”(代理备案号 No. 06005/591623 和 No. 20040/59-11623)；“用于过程控制系统的集成图形运行时界面”(代理备案号 No. 06005/591628 和 No. 20040/59-11628)；“面向服务的过程控制系统架构”(代理备案号 No. 06005/591629 和 No. 20040/59-11629)。

### 技术领域

[0003] 本发明总的来说涉及加工工厂，更具体而言，涉及过程控制和仿真系统的系统级图形显示编辑器和图形显示对象的集成和使用，进而使得公用图形显示元素能够在各种与工厂配置、控制、维护和仿真有关的活动中创建和使用。

### 背景技术

[0004] 分布式过程控制系统，比如用在化学、石油或其他过程中的分布式过程控制系统，通常包括一个或更多过程控制器，这些过程控制器通过模拟、数字或模拟数字混合总线，连接到一个或更多现场设备上。所述现场设备可以是阀门、阀门定位器、开关和变送器（例如，温度、压力、液位和流速传感器），它们位于过程环境内，并执行比如开/关阀门、

测量过程参数等功能。智能现场设备,例如符合公知的 Fieldbus(现场总线)协议(比如 FOUNDATION™Fieldbus 协议)的现场设备,也可以执行控制计算、报警功能以及其他通常在控制器中实现的控制功能。过程控制器通常也位于工厂环境内,可以接收表示由现场设备所做的过程测量结果和/或关于现场设备的信息的信号,并可以执行一个控制器应用程序,例如该应用程序运行不同的控制模块,这些控制模块可以基于所收到的信息做出过程控制决策、生成控制信号,以及与在比如 HART、Fieldbus 现场设备之类的现场设备中正在执行的控制模块或者块协同工作。控制器中的控制模块通过通信线路将控制信号发送到所述现场设备,从而控制该过程的操作。

[0005] 来自现场设备和控制器的信息通常可通过数据总线应用于一个或更多其它位于控制室或其他远离嘈杂的工厂环境的硬件设备,例如操作员工作站、个人计算机、数据历史记录站、报告发生器、集中数据库等。在这些硬件设备上运行应用程序,这些应用程序比如使操作员能够执行与过程有关的功能,例如改变过程控制例程的设置、修改控制器或现场设备中控制模块的操作、查看过程的当前状态、查看由现场设备和控制器产生的警报、仿真过程的操作以培训人员或测试过程控制软件、维护和更新系统配置数据库等。

[0006] 例如,由爱默生过程管理出售的 DeltaV™ 控制系统包含多种应用程序,这些应用程序存储在加工厂内位于多个地方的不同设备中,并由加工厂内位于多个地方的不同设备来执行。配置应用程序存在于一个或更多操作员工作站中,并使得用户可以创建或更改过程控制模块,并通过一个数据总线将这些过程控制模块下载到专用分布式控制器。通常,这些控制模块由通信上互连的功能块组成,这些功能块是面向对象编程协议中的对象,这些对象基于输入执行控制方案内的功能,并且将输出结果提供给控制方案内的其他功能块。配置应用程序还允许设计人员创建或改变操作员界面,这些操作员界面由一个浏览应用程序所使用,来向操作员显示数据并使得操作员能够在过程控制例程内更改设置,例如设置点。每个专用控制器以及某些情况下的现场设备,存储并执行一个控制器应用程序,该应用程序运行所分配和下载的控制模块来实现实际的过程控制功能。可以在一个或更多操作员工作站上运行的浏览应用程序,可以通过数据总线接收来自控制器应用程序的数据,并将所接收数据显示给过程控制系统设计者、操作员或者使用用户界面的用户,还可以提供任意多个不同视图,例如操作员视图、工程师视图、技术员视图等等。数据历史应用程序通常存储于一个数据历史设备中并由该设备来执行,该数据历史设备收集和存储通过数据总线得到的部分或全部数据,同时可以在一个与数据总线连接的计算机上运行一个配置数据库应用程序,来存储当前过程控制例程配置和与之相关联的数据。作为选择,配置数据库也可以与配置应用程序放置在同一个工作站中。

[0007] 鉴于用在过程控制环境中的控制和支持应用程序的数量和种类都有所增加,为了使用户能够有效地配置和使用这些应用程序,已经有不同种类的图形显示应用程序问世。例如,为了使配置工程师能够通过图形来创建控制程序,且这些程序能够下载到加工厂的控制器设备上,图形显示应用程序已经被用来支持控制配置应用程序。另外,图形显示应用程序也已经被用来达到以下目的,例如,使操作人员能够查看加工工厂当前的运行情况或者加工厂的各个区域,使维护人员能够查看加工厂内硬件设备的状态,能够启动加工厂的仿真,等等。然而在过去,这些图形显示应用程序是作为其所关联具体应用程序一部分被创建的,或者是为了支持其所关联具体应用程序而创建的,因此一般在使用性方面具有局限性,

局限于其被创建时的具体过程功能。例如,很难或几乎不可能将一个被创建用于支持控制或其他操作员的图形程序应用于维护、配置或者仿真功能。

[0008] 举一个具体的例子,一些过程控制配置应用程序目前包含一个模板对象库,例如功能块模板对象,或者在某些情况下的控制模块模板对象,这些对象用于为加工厂创建一个控制策略。模板对象具有默认的属性、设置和与其关联的方法,而且使用图形配置应用程序的工程师可以选择这些模板对象,然后实质上将所选模板对象的复制件放置在配置屏幕上,以开发控制模块。在选择和放置模板对象于配置屏幕的过程中,工程师将这些对象的输入和输出相互连接,并更改这些对象的参数、名字、标签和其他属性,以创建一个在加工厂中具有特定用途的特定控制模块。在创建一个或更多这样的控制模块后,工程师接着可以实例化该控制模块,并将该控制模块下载到合适的一个或更多控制器以及现场设备上,以便在加工厂运行期间执行。

[0009] 其后,工程师可以使用一个不同的图形显示创建应用程序,通过在显示创建应用程序中选择和建立显示对象,为在加工厂内的操作员、维护人员等创建一个或更多显示。这些显示通常是在一个或更多工作站基于泛系统实现的,并能够为操作员或维护人员提供预配置显示关于控制系统或工厂内设备的运行状态。这些显示的形式通常包括:报警显示,所述报警显示接收并显示控制器或加工厂内设备所生成的警报;控制显示,指示控制器或加工厂内其它设备的运行状态;维护显示,指示加工厂内设备的功能状态,等等。然而,这些显示一般会预先配置,从而以已知的方式来显示从过程控制模块或加工厂内设备所接收到的信息或数据。在有些系统中,由一个图形描绘来创建显示,该图形描绘代表一个物理或逻辑单元,并与所述物理或逻辑单元之间存在通信连接,从而可以接收关于该物理或逻辑单元的数据。显示屏幕上的图形可以响应某些事件而改变,例如所接收的数据表明比如说有一个储罐半满,表明流量传感器所测量的流量,等等。但是,用于配置、操作员控制、维护和仿真活动的图形显示一般都是使用不同的图形编辑器彼此单独创建的。另外,这些显示有限的图形能力很难也不能实现任何图形对象的一部分。

[0010] 鉴于此,类似于控制配置应用程序,显示创建应用程序可以具有模板图形显示项,比如储罐、阀门、传感器、滑杆之类的操作员控制按钮、开关等等,这些项可以位于任何期望配置的屏幕上来创建操作员显示、维护显示以及其他类似显示。当把个别图形项放在屏幕上时,这些项可能在屏幕上相互连接,其连接方式可以为用户提供加工厂内部工作的一些信息或显示。但是,为了将图形显示做成动画,显示创建者必须通过在图形项和加工厂内的相关数据源之间指定通信链接,来手工将每个图形项与加工厂内生成的数据结合在一起,所述加工厂内生成的数据比如是传感器测量到的数据或者阀门定位器指示的数据,等等。这样的处理过程非常冗长耗时,而且可能会充满错误。

[0011] 虽然控制配置应用程序中的控制模板对象和显示创建应用程序中的显示项比较方便,因为它们可以被复制和用于创建多种不同的控制模块和图形显示,但是经常需要为加工厂内不同设备创建大量同样的控制模块和图形显示。例如,很多情况下,中型到大型加工厂都有同样的或相似的设备,这些设备可以用同样的基本通用控制模块和显示来控制 and 查看。然而,为了创建这些大量的控制模块和显示,可以创建一个通用控制模块或显示模块,然后为每个可以使用所述通用控制模块或显示模块的不同设备都复制所述通用控制模块或显示模块。当然,在复制之后,必须在配置应用程序中手工更改每个新的控制或显示模

块,从而指定某个设备附着于哪个新的控制或显示模块,然后将所有这些控制和限制模块必须实例化并下载到过程控制系统。

[0012] 不幸的是,上述控制模块和显示项并非在任何形式下都是模块化的。因此,在复制操作后,必须使用合适的配置应用程序来手工逐个更改所述控制模块和显示,从而指定工厂内的设备与哪个控制模块和显示相关联。对于一个工厂,如果它具有同一类型设备的很多拷贝(例如被复制的设备),这个过程将会冗长耗时,并充满人为操作错误。而且,这些不同的控制模块和显示一旦被编程,这些控制模块和显示将互不察觉。因此,为了在创建控制模块时进行某种改变,工程师或操作员必须手工对不同复制设备的每个不同控制模块进行同样的改变,这种操作同样也是冗长耗时的。同样的问题也存在于工厂内不同套的复制设备所创建的图形视图。换句话说,一旦创建了一个具体控制模块或者一个具体的图形视图(分别建立或者从一个模板对象进行复制),然后将所建控制模块或图形视图连接到工厂内的一套特定设备,所述控制模块或图形视图是系统内的独立实体或对象,即使系统内的其他控制模块或图形视图与所述控制模块或图形视图相同或相似。结果,适用于每个特定类型的控制模块和图形显示的变化,对于这些模块和显示来说必须是单独进行的。当图形显示是为工厂内不同功能背景中,比如对于控制查看、维护查看和仿真功能的相同设备创建时,这个问题甚至更加明显。在这种情况下,就要单独创建图形视图,而这些视图相互间并不知道对方的存在。

[0013] 因此,虽然图形视图已经被提供给和关联到不同应用程序,其中所述应用程序用于加工工厂内进行的不同常规活动,但是这些图形显示和关联的图形显示编辑器一般被添加到功能级的应用程序,而所述图形显示和编辑器正是用于支持这些应用程序而创建的。结果,图形编辑器在一定程度上只是使用户能够建立一些视图,这些视图能够支持特定应用程序所需的特定功能。以前的加工工厂没有提供这样的图形显示编辑器,该编辑器用于或者支持在工厂配置和支持范围内各种活动的图形需要。从而,举例来说,一个用于支持或者启动控制配置活动的图形显示编辑器只能使用户创建控制程序,而不支持操作员或者维护显示的需求或功能。同样地,用于创建操作员视图、维护视图等(其中所述视图在工厂操作过程中被提供给控制操作员或者维护技师)的图形显示编辑器不支持与配置活动、仿真活动等关联的功能。由于需要在加工工厂的各个功能级,例如在控制配置、维护支持、控制操作员支持和仿真支持功能级,支持图形显示,由各种编辑器所创建的不同显示最终模拟和描述工厂内的相同部件,结果导致各种不同的工作人员在加工工厂的图形显示方面加倍付出努力。所述加倍努力不仅表现在需要建立描述不同用途的相同过程元素的不同图形显示,而且表现在将用于不同显示应用程序的图形元素连接到加工工厂内所关联的实际硬件或软件单元。

[0014] 由于之后已经提供了对于各种加工工厂活动的图形支持,并且作为实际执行活动的一部分,图形支持没有被集成在工厂环境中来允许创建通用图形并在工厂的各种不同功能级处进行使用。这种图形非集成引发了针对不同功能或者不同应用的不同功能的图形,从而导致用户侧的疑惑,因为用户可能偶尔需要不时地查看与工厂内不同操作或功能相关联的不同显示,而用户一般只熟悉一种特定类型的图形显示。同样,正如上面所提到的,对工厂内各种不同功能级图形显示的支持,会导致图形支持的加倍,无论是在创建显示方面,还是在将显示内的元素适当地连接到工厂内实际硬件和软件单元方面。

[0015] 另外, 差错检测和其他编程对于检测与不同控制器中运行的控制回路相关的条件、错误、警报等, 以及单个设备内的问题都很有用。传统上来说, 这种差错检测在加工工厂的不同功能级处执行, 并被显示在为那些不同功能活动的图形显示上。因此, 一直都很难对过程控制系统编程来识别系统级的状况和错误, 其中这些状况和错误必须通过分析广泛分布在加工工厂内的不同设备的数据才能检测到, 而且, 将这些类型的错误显示在操作员显示上更加困难, 因为还没有建立将这种系统级的状况信息显示或呈现给操作员或维护人员的操作员显示。而且, 也很难用显示内不同元素的这些交替信息源或数据源将操作员显示内的对象做成动画。

## 发明内容

[0016] 提供用于过程环境中的图形元素和图形显示, 以便将与过程环境相关的信息例如加工工厂内设备的当前状态显示给一个或更多用户。在配置加工工厂过程中, 可以将图形元素和图形显示与加工工厂内各种逻辑和物理单元关联, 也可以将图形元素和图形显示与其他配置项目例如控制例程一同配置并下载到加工工厂内的硬件上。特别是, 可以创建图形元素和图形显示并将其存储在库中, 然后通过将其与加工工厂内的各种逻辑和物理单元关联来对图形元素和图形显示进行配置。在配置过程期间, 图形元素和图形显示按工厂配置中其它处所定义的, 可以与工厂的区域、设备、处理模块、控制例程或者控制策略关联, 或者可以与接口或显示设备关联, 从而定义在运行时间内在其上执行图形显示的硬件, 并进而定义与图形显示绑定的过程实体, 以实现显示的目的。更进一步, 可以为每个图形显示定义一角色或功能用途, 例如操作员视图、维护视图等等, 而且这些角色可以用于定义在运行时环境内对图形显示适当访问和使用。

## 附图说明

[0017] 图 1 是位于加工工厂内的分布式过程控制网络的框图, 该加工工厂包括执行显示例程和与加工工厂内各项功能相关联的其他应用程序的操作员工作站, 以及提供系统级图形支持的工作站, 所述系统级图形支持可以用来创建图形显示元素和用于工厂各个功能区域的图形显示;

[0018] 图 2 是图示集成加工工厂控制、查看和仿真系统内系统级图形支持的逻辑框图;

[0019] 图 3 是图示创建图形元素和显示的配置环境, 以及可以执行图形元素和显示的运行时环境的逻辑图;

[0020] 图 4 是由图形编辑器产生的简化显示屏幕, 能够使用户创建或编辑泵元素形式的图形元素;

[0021] 图 5 是当产生反应器形式的图形元素时图形编辑器产生的另一显示屏幕;

[0022] 图 6 的框图图示各种显像可以捆绑到或关联到图形元素上的一种方式;

[0023] 图 7 描绘屏幕显示的一部分, 它展示与图形元素相关联的第一属性视图;

[0024] 图 8 描绘屏幕显示的一部分, 它展示与图形元素相关联的第二属性视图;

[0025] 图 9 描绘显示屏幕的动作 / 动画部分, 它展示与图形元素属性相关联的动作;

[0026] 图 10 的框图图示了可视触发器与图形元素的属性和显像集成在一起的一种方式;



- [0027] 图 11 描绘可以用来为图形元素的显像提供或定义变换动画的对话框；
- [0028] 图 12 描绘了可以用来为图形元素的显像提供或定义属性动画，包括颜色动画的对话框；
- [0029] 图 13 的图概括地通过解析表将图形元素捆绑到运行时环境的方式；
- [0030] 图 14 的框图描绘了绑定到过程环境中多个不同数据源的显示元素；
- [0031] 图 15 是当产生来自多个显示元素和连接器的图形显示时，由图形编辑器产生的显示屏幕；
- [0032] 图 16 是用于创建图形显示的显示屏幕，该图形显示图示了与置于该图形显示内的图形元素相关联的各种显像 (visualization)；
- [0033] 图 17 是与由各种互连图形元素和连接器组成的图形显示相关联的显示屏幕；
- [0034] 图 18 是可以经由图 17 的图形显示进行访问的一组图形显示屏幕，用来展示用于图 17 显示内的元素的控制面板和面板；
- [0035] 图 19 是图形显示的显示屏幕，它图示了由各种显示元素组成的石灰窑，并提供来自加工厂内各个其他应用程序和数据源的数据；
- [0036] 图 20A ~ 20E 的图形显示屏幕图示了具有相同观感的石灰窑的不同视图，包括操作员视图、工程视图、仿真视图和维护视图；
- [0037] 图 21A 和 21B 是与用来运行图 20A~20E 所示石灰窑的控制例程相关联的显示屏幕；
- [0038] 图 22 的框图图示了处于不同级别，包括物理级和功能级的各种不同图形显示之间的关联；
- [0039] 图 23 描绘了第一配置屏幕，它图示了与配置系统相关联的图形配置分级结构；
- [0040] 图 24 描绘了第二配置屏幕，它图示了一个图形配置分级结构，该分级结构展示了图形元素和图形显示可以分配给该配置系统内的其他元素，并与之集成在一起的一种方式；和
- [0041] 图 25 描绘了一个对话框，它可以提供给用户用来总结在加工厂配置系统内分配一个或更多图形显示的方式。

### 具体实施方式

[0042] 图 1 图示了一个示例性加工厂 10，其中将系统级图形支持提供给工厂 10 的各个功能区域。典型地，加工厂 10 包括具有一个或更多控制器 12 的分布式过程控制系统，其中每个控制器 12 均经由输入/输出 (I/O) 设备或卡 18 与一个或更多现场设备 14 和 16 相连，所述输入/输出设备或卡例如可以是 Fieldbus 接口、Profibus 接口、HART 接口、标准的 4-20 毫安接口等等。控制器 12 还可以经由数据总线连接至一个或更多主机或操作员工作站 20-23，所述数据总线例如可以是以以太网链路。数据库 28 可以连接至数据总线 24，并作为数据历史记录器来操作，以采集和存储与工厂 10 内的控制器和现场设备相关联的参数、状态和其他数据，和/或作为配置数据库来操作，该配置数据库存储下载到并存储在控制器 12 和现场设备 14 和 16 内的、工厂 10 内过程控制系统的当前配置。数据库 28 另外还可以存储以此处描述的方式创建的图形对象，以提供加工厂 10 内的图形支持。尽管控制器 12、I/O 卡 18 和现场设备 14 和 16 通常位于并且遍布于整个有时候很恶劣的工厂环境内，

操作员工作站 20-23 和数据库 28 通常位于控制室内,或者其他容易由控制器或维护人员进行访问的不太恶劣的环境中。然而,在某些情况下,手持式设备可以用来执行这些功能,并且这些手持式设备通常可以携带到工厂中的各个地方。

[0043] 众所周知,每个控制器 12,例如可以是由艾默生过程管理 (Emerson Process Management) 公司销售的 DeltaV™ 控制器,均存储和执行控制器应用程序,该控制器应用程序利用任意数目的不同的独立执行的控制模块或控制块 29 来实现控制策略。每个控制模块 29 均可以由通常所称的功能块组成,其中每个功能块均作为整个控制例程的一部分或子例程,并且连同其他功能块一起共同操作(通过称作链接的通信)以便实现加工厂 10 内的过程控制回路。众所周知,可以是面向对象编程协议中的对象的功能块,通常执行以下功能中的一种,例如与变送器、传感器或其他过程参数测量设备相关联的输入功能,例如与执行 PID、模糊逻辑等控制的控制例程相关联的控制功能,或者控制某些设备如阀门的操作以执行加工厂 10 内某些物理功能的输出功能。当然,还存在混合及其他类型的复杂功能块,例如模型预测控制器 (MPC),优化器,等等。尽管 Fieldbus 协议和 DeltaV 系统协议使用以面向对象编程协议设计和实现的控制模块和功能块,但是控制模块也可以利用任何期望的控制编程方案来设计,包括例如顺序函数块,梯级逻辑等等,但不限于利用功能块或任何其他特定编程技术进行设计和实现。

[0044] 在图 1 所示的工厂 10 中,与控制器 12 相连的现场设备 14 和 16 可以是标准的 4-20 毫安设备,可以是智能现场设备,例如包括处理器和存储器的 HART、Profibus 或 FOUNDATION™ Fieldbus(现场总线)现场设备,或者可以是任何其他期望类型的设备。这些设备中的某些,例如 Fieldbus 现场设备(在图 1 中用附图标号 16 标注),可以存储和执行与在控制器 12 中所实施控制策略相关联的模块或子模块,例如功能块。众所周知,在图 1 中图示为布置于两个不同 Fieldbus 现场设备 16 中的功能块 30,可以连同控制器 12 内控制模块 29 的执行一起来执行,以实现过程控制。当然,现场设备 14 和 16 可以是任意类型的设备,例如传感器、阀门、变送器、定位器等等,而 I/O 设备 18 可以是符合任何期望的通信协议或控制器协议如 HART、Fieldbus、Profibus 等的任意类型的 I/O 设备。

[0045] 在图 1 的加工厂 10 中,工作站 20-23 可以包括各种应用程序,这些应用程序可用于由工厂 10 内相同或不同人员执行的各种不同功能。工作站 20-23 中的每一个均包括存储器 31 和处理器 32,该存储器 31 用于存储各种应用程序、程序、数据结构等,该处理器 32 可以用来执行存储在存储器 31 中的任一应用程序。在图 1 所示的例子中,工作站 20 被标明为配置工作站,并且包括一个或更多配置应用程序 33,所述配置应用程序例如可以包括控制模块创建应用程序、操作员界面应用程序,以及可以由任何经授权的配置工程师进行访问,以创建和下载控制例程或模块如控制模块 29 和 30 至工厂 10 的各个控制器 12 和设备 16 的其他数据结构。工作站 21 在图 1 中一般图示为控制操作员查看工作站,并且包括若干显示应用程序 34,该显示应用程序可以向控制操作员提供加工厂 10 运行期间的各种显示,以便使操作员能够查看和控制加工厂 10 内或工厂各个部分中所发生的一切。应用程序 34 可以包括支持应用程序 34a,例如控制诊断应用程序、调谐应用程序、报告生成应用程序,或可以用来辅助控制操作员执行控制功能的任何其他控制支持应用程序。类似地,工作站 22 图示为维护查看工作站,并且包括若干维护应用程序 35,该维护应用程序可以由各种维护人员用来查看工厂 10 的维护需求,查看各个设备 12、14、16 等的操作或工作状态。当然,应用

程序 35 可以包括支持应用程序 35a, 例如维护诊断应用程序、校准应用程序、振动分析应用程序、报告生成应用程序, 或可以用来辅助维护人员执行工厂 10 内维护功能的任何其他维护支持应用程序。另外, 工作站 23 图示为仿真工作站, 它包括若干仿真应用程序 36, 该仿真应用程序可以用来为了各种目的, 例如为了训练目的, 为了工厂建模的目的, 来仿真工厂 10 或工厂 10 各个部分的操作, 以辅助工厂维护和控制, 等等。通常, 工作站 20-23 中的每一个均包括显示屏 37, 以及其他标准外围设备, 如键盘、鼠标, 等等。

[0046] 当然, 尽管在图 1 中将各种配置、控制、维护和仿真应用程序 33-36 图示为位于专用于这些功能之一的不同工作站中, 但是应当理解, 与这些或其他工厂功能相关的各种应用程序 33-36 可以位于工厂 10 内的相同或不同工作站或计算机中并由其执行, 这取决于工厂 10 的需求和设置。因此, 举例来说, 一个或更多仿真应用程序 36 和控制应用程序 33 可以在工作站 20-23 当中的同一个工作站中执行, 而不同的个别仿真引用程序 36 或不同的个别控制应用程序 33 可以在工作站 20-23 当中的不同工作站中执行。

[0047] 在过去, 相当独立地执行工厂 10 不同功能区域中使用的不同应用程序的开发。因此, 配置应用程序 33 的开发不与仿真应用程序 36、维护应用程序 35 或操作员控制应用程序 34 集成在一起。实际上, 在许多情形下, 工厂可能已经包含了由不同公司或软件提供商开发的、用于不同功能区域的应用程序, 并且这些应用程序实际上被开发为独立于工厂 10 内的其他软件而运行。由于与工厂 10 内各个功能区域相关联的不同应用程序的这种独立开发和运行, 一般要求工厂人员在配置、操作员控制、维护和仿真功能级的每一级分别配置或设置工厂。特别是, 相同或不同的工厂人员一般不得不使用不同的程序, 以在各个功能级建立新的数据结构和图形显示。因此, 对于图 1, 执行配置、控制、维护和仿真功能的各个应用程序 33-36 中的每一个, 一般均包括或使用不同的图形显示编辑器和数据结构, 以辅助工厂人员执行这些配置、操作员控制、维护和仿真功能。在许多情形下, 这些不同的图形显示编辑器和数据库用来创建不同的图形显示, 以便对工厂 10 的相同部分或区域, 或工厂 10 内的相同硬件进行描绘或建模, 从而辅助不同的工厂人员来可见和理解在配置、操作员控制、维护或仿真活动的场景下加工工厂内所发生的一切。

[0048] 由于一般是彼此独立地, 有时候由不同的人员甚至不同的公司来开发和执行应用程序 33-36 以及用于工厂 10 内每个不同功能的相关显示, 因此, 从提供图形显像的立场来看, 在加工工厂不同功能区域中创建或使用的图形显示, 并不以任何一致的或容易理解的方式集成。因此, 图形显示在工厂各个不同功能级的独立创建和执行, 导致这些图形显示看起来每个功能都彼此不同, 从而跨越整个功能区域上的图形显示没有一致的观感。另外, 这种独立创建导致为工厂的相同部分或区域但不同的功能用途创建图形显示的精力加倍, 并且要求如此创建的图形显示分别在工厂 10 的各个功能级捆绑到工厂 10 内的各个设备, 例如控制器 12 和现场设备 14, 16, 或者接收来自工厂 10 内各个设备的数据。该事实进而又要求数据结构的重复, 以便为不同的显示跟踪相同的硬件单元。因此, 举例来说, 在过去, 第一应用程序 (例如一个应用程序 35) 用来创建维护显示, 该维护显示是为了维护目的来图示工厂 10 的一部分, 而第二应用程序 (例如一个应用程序 34) 用来创建控制操作员显示, 该控制操作员显示是为了控制目的来图示工厂 10 的相同部分。由这些不同显示编辑器分别创建的显示在观感上可能差异相当大, 这使得用户难以在维护显示和操作员显示之间向后和向前切换, 并且不易混淆或者无须要求关于各种类型的显示的训练。同样, 在不同的应用程

序 34 和 35 中独立地创建两个显示,也使所花费的精力加倍,并且增加了额外的精力来创建数据结构以便分别将维护显示和控制操作员显示捆绑或连接到工厂 10 内的相同硬件单元上,进而从那些硬件单元接收有时相同或类似的数据。

[0049] 为了减轻这些低效率的工作,并且为了在工厂 10 内提供可以更广泛使用且更容易理解的图形,在加工工厂 10 的系统级配备图形支持层以支持工厂 10 各个功能区域中的每一个,包括工厂 10 的配置、操作员查看、维护查看、仿真和其他功能区域的图形显示和数据结构需求。在图 2 中用图解法描绘了该系统级支持,图 2 图示了工厂操作级 40、工厂功能级 42 和系统级 44。通过图 2 可以理解,工厂操作级 40 包括控制器 12、现场设备 14,16 等,它们执行控制例程或模块 29 和 30 以及在工厂 10 内运行的其他软件,以实施工厂运行时间期间的工厂操作。将工厂功能级 42 描绘为包含配置功能块 46、控制功能块 47、维护功能块 48 和仿真块 49,但是也可以配备其他或不同的功能,例如工程和商业功能。配置功能块 46 执行配置例程 33,配置例程 33 与工厂操作级 40 内的部件进行连接或通信,以向其提供控制策略或控制模块。控制功能块 47 包括控制查看和其他应用程序 34 和 34a,它们通常也直接与工厂操作级 40 内的各个物理和逻辑部件进行连接或通信,以便执行工厂 10 内操作员发起的变化,经由控制显示 34 向操作员提供信息,获取用于控制应用程序 34a 的数据,等等。维护功能块 48 包括维护例程和应用程序 35 和 35a,它们与工厂操作级 40 内的各个物理和逻辑部件进行连接和通信,以便执行维护程序,采集维护数据,经由维护显示 35 向维护人员提供维护数据或信息,运行诊断应用程序 35a,等等。同样,仿真功能块 49 包括仿真例程 36,该仿真例程 36 执行工厂 10 的仿真,并且可以与工厂操作级 40 内的部件通信连接,以获得关于工厂 10 的数据。

[0050] 如图 2 所示,系统级支持层 44 被捆绑到工厂功能层 42 内的每个功能块 46-49 中,并支持工厂功能层 42 内的每个功能块 46-49,以便实现诸如公用数据库和显示结构,如用于各个功能区域 46-49 的软件对象、图形元素 (graphic element) 和图形显示的创建和维护。更具体而言,系统级支持层 44 包括应用程序、数据库和图形支持元素,它们能够使在各个功能块 46-49 中执行的图形活动集成到一起,或者利用在系统支持层 44 创建的公用数据库结构和图形元素进行开发。为了提供该系统级支持,系统支持层 44 可以包括图形编辑器 50 和图形对象数据库 52。图形编辑器 50 可以用来创建图形元素 54 和图形显示 56,而图形对象数据库 52 将图形元素 54 和图形显示 56 存储在可以由编辑器 52 和功能块 46-49 中各个应用程序进行访问的存储器中。数据库 52 还可以存储其他对象 58,以及将图形元素 54 连接至工厂操作级 40 内各个硬件和软件单元的数据结构。另外,数据库 52 可以存储可以用来创建更多图形元素或显示的图形元素或显示模板或图元。通过图 2 可以理解,图形元素 54、图形显示 56 和其他数据库结构 58 可以由任意一个和所有功能块 46-49 用来创建和使用与这些功能块相关的图形。

[0051] 一般而言,系统级支持层 44 提供一种将图 1 的加工工厂 10 中使用的图形在所有功能区域 46-49 进行集成的方式,由此降低和消除在不同工厂场境下为相同的工厂设备重复创建不同图形元素的必要性,并且使各个功能区域 46-49 的用户容易联系到与在那些功能区域相关联的图形视图中显示的设备相关联的数据。应当理解,系统级支持层 44 可以用来为各个功能区域 46-49 中的多个应用程序,为不同功能区域 46-49 中的不同应用程序提供图形和数据库支持。

[0052] 再次参见图 1, 系统级支持块 44 可以利用与其他工作站 20-23 中任一相连的另外的工作站或用户接口 60 来实现。工作站 60 一般可以存储图形编辑器 50 和数据库 52, 如果需要的话, 也可以存储其他元素 54、56 和 58。另外, 工作站 60 可以经由数据总线 24, 经由分离的有线和无线通信连接 (在图 1 中用虚线表示), 或者以任何其他期望的方式与工作站 20-23 通信连接。在图 1 所示的配置中, 工作站 60 存储和执行显示编辑器 50, 以便使用户能够创建图形元素, 并将这些图形元素分为一个或多个图形显示组, 此处这两组都称为显示模块。然后可以将这些显示模块存储在数据库 52 中, 以便由图 2 所示的各个功能块 46-49 进行访问和使用, 并且在各个工作站 20-23 上执行。尽管为了图解说明起见, 将系统级块 44 和功能级块 46-49 的功能性图示为在图 1 的不同或各个工作站 20-23 和 60 上执行, 但是应当理解, 与这些不同块中任一相关联的任一或所有应用程序都可以在相同或不同的工作stations 上, 或者在加工厂 10 内或与加工厂 10 相关联的其他计算机上执行。因此, 图形编辑器 50 可以存储在任一其他工作站 20-23 或者与工厂 10 相关联的任何其他计算机中, 并且在任一其他工作站 20-23 上或者在与工厂 10 相关联的任何其他计算机上执行, 并且无须是独立计算机或孤立计算机。

[0053] 如上所述, 图 2 的系统级层 44 执行系统级显示对象和数据库对象, 它们可以在各种功能环境中使用, 并且提供更高级的显示能力。一般而言, 在图 2 的系统级 44 创建的显示对象可以归类为图形元素和图形显示。图形元素一般是与工厂内特定物理实体相关联的图元或低级显示对象, 该特定物理实体例如为像阀门、传感器、泵、控制器等的硬件设备。图形显示一般由一组互连的图形元素组成, 并且对工厂内更复杂的各组硬件如单位、区域等进行表示和建模, 它包括不同硬件单元之间的相互联系。另外, 图形显示可以包括图形, 图表, 以及从工厂, 从其他应用程序如在工作站 20-23 和 60 等上运行的诊断和商业应用程序提供的其他数据。

[0054] 图 3 概括地图示了图形元素和图形显示在可能存在这些图形元素和图形显示的两种环境下的开发和使用的, 特别是配置环境 70 和运行时环境 72。一般而言, 在配置环境 70 中利用例如显示编辑器 50 来创建图形元素 74 (描绘为单独的元素对象 74a, 74b 等) 和图形显示 76 (描绘为单独的显示对象 76a, 76b 等) 形式的显示对象。在创建之后, 对象 74 和 76 可以存储在数据库 52 中。可以将对象 74 和 76 创建为类对象, 此处称作显示模块类对象, 它定义了未绑定或捆绑到加工厂 10 内特定硬件或逻辑单元的分类对象。然而, 类对象可以用来创建具有与类对象相同的基本属性, 但捆绑或绑定到加工厂 10 内特定硬件的运行时图形对象。然而, 一般而言, 类对象仍然捆绑到由其实例化的子对象上, 从而即使当在运行时环境内实例化这些子对象时, 类对象的变化也可以自动地传播给了对象。

[0055] 如图 3 所示, 每个图形元素对象 74 均包括使图形元素在众多不同场景中用的大量部件。特别是, 每个图形元素 74 均包括一个或更多显像 77、任意数目的参数或属性 78、任意数目的可以利用脚本或触发器执行的动作或动画 79、和绑定 80。一般而言, 每个显像 77 均定义了当在运行时环境 72 中实现图形元素 74 时要在显示屏上实际显示的可视属性或元素。通常, 显像定义一个物理或逻辑设备, 或一组设备的表示, 但显像也可以表示其他实体。可以利用定义了实体的图形描绘细节的、任何期望的描述或程序设计范例在运行时环境 72 中实现显像 77。在一个实施例中, 显像 77 可以利用 PGXML 或 Avalon 控件来实现, 执行控件是由微软 Microsoft<sup>®</sup> 提供的知名控件, 并且由于它们是基于对象的, 因此这些控

件提供这样一种方式,使显像能够容易地在标准Windows®型显示中实现,并且容易在显示环境之间可移植。该特征将在下文中进行更详细的讨论,并且在标题为“加工厂用户界面中基于标记语言的动态过程图形 (Markup Language-Based, Dynamic Process Graphics in a Process Plant User Interface)” (代理档案号 No. 06005/41127) 的共同未决申请中进行解释说明,该申请合并于此以供参考。

[0056] 一般而言,参数和属性 78 定义与由显像描绘的实体相关联的变量或其他属性,例如静态或可变的固有属性,并且这些属性可以由图形元素 74 的创建者进行定义。同样,动作和动画 79 定义例程或程序 (可以作为执行属性交换,基于属性值检测过程实体状况等的脚本来执行),定义动画例程,该动画例程可以包括当在显示屏幕上描绘显像 77 时改变要在显像 77 上执行的或者利用显像 77 执行的图形显像或行为的任何例程,或者能够使用户利用显像 77 或者与显像 77 交互从而引起过程变化例如改变过程输入的例程。这些动作和动画为显像 77 提供更加有趣、容易理解或有用的图形属性,并且允许用户与显像 77 交互。在一种情况下,这些动作或动画可以采取各种形式,如显像各个部分的颜色、尺寸 (例如高度和宽度、线尺寸、字体等) 变化,颜色填充变化,和动画变化,例如颜色、旋转变化,尺寸和比例、偏移变化,等等。这些动作和动画向图形元素 74 提供图形属性,以及用户交互属性。绑定 80 可以是静态或固定绑定或者使用别名的绑定,它定义这样一种方式:当作为运行时环境 72 中显示的一部分实现图形元素 74 时,要将参数或属性 78 绑定到运行时环境 72 内的数据、标签或其他实体上。基本上,用于各个图形元素 74 的绑定 80 建立这样一种方式,通过该方式将图形元素 74 捆绑到在工厂环境中其他地方定义的一个或更多实体或数据元素上,由此定义实际运行时环境 72 和图形元素 74 之间的接口。

[0057] 如图 3 所示,每个图形显示对象 76 均包括许多部件,例如对一个或更多图形元素 81、连接器元素 82、动作和动画 83、属性 84 和绑定 85 的引用或副本。一般而言,图形显示 76 可以是描绘各个图形元素 81 的交互的显示,所述各个图形元素 81 可以可视化地与代表管道、线路、传送带等的连接器元素 82 连接到一起。在美国公开文 No. 2004/0153804 中描述了这样的连接器对象。图 3 中的虚线表示图形显示对象 76a 对一个图形元素 74 的引用。应当理解,引用图形元素 74 的图形显示 76 包括该图形元素 74 的所有属性、参数、动作和动画,等等。与图形元素 74 类似,每个图形显示 76 均可以包括一个或更多与之相关联的另外的动作或动画,这些动作或动画执行例如显示上的动画、用户界面交互、数据操控,等等。同样,每个图形显示 76 均可以包括与该显示相关联的任意数目的属性,并且通常这些属性定义该显示内描绘的单位、区域、或其他组元素的属性。当然,绑定 85 定义这样一种方式,通过该方式将图形显示 76 捆绑到在工厂环境中其他地方定义的一个或更多实体或数据元素上,由此定义实际运行时环境 72 和图形显示 76 之间的接口。

[0058] 一旦创建图形元素 74 和图形显示 76,它们可以在运行时环境 72 下绑定到例如如图 1 的任一工作站 20-23 上,并且在图 1 的任一工作站 20-23 上执行。特别是,在将图形元素 74 或图形显示 76 创建为类对象并存储在数据库 52 中之后,该图形元素或图形显示可以实例化为实际运行时对象,并且可以在运行时环境 72 中执行。如框 86 所示,实例化过程填充到在对象 74 和 76 中定义的绑定中,这可以利用一个或更多解析表来完成,所述解析表可能装载有加工厂或过程控制系统内适当的名称、标签、别名等,以便提供加工厂内实际实体与在工厂 10 内显示设备上运行的图形对象之间的专用连接。作为绑定过程的一部分,对象 74

和 76 连接至由解析表定义的加工厂内的数据源,由此获得对工厂的访问,以便在逻辑上并且可通信地连接至加工厂 10。

[0059] 如框 87 所示,显示元素 74 或图形显示 76 可以在运行时环境 72 内的若干不同功能中执行,或者作为若干不同功能的一部分执行,所述功能包括配置显示、控制操作员显示、维护显示和仿真显示,仅举这几个为例。另外,显示对象 74 和 76 可以用于执行系统级功能,例如使用来自图 2 所描绘的各种功能级的数据的功能,包括例如预测控制或预测维护功能、系统级检修、诊断等。实际上,显示 76 一旦在配置环境 70 中创建并存储在数据库 52 中,显示 76 就可以用于若干不同的活动。更进一步,显示对象 74 和 76 可以在任何期望的显示器或计算机硬件上执行,例如工作站 90、膝上型计算机 91、手持式设备 92、像个人数字助理 (PDA)、电话设备等,或者任何其他专业显示器 93,例如具有多个监视器的大屏幕显示器,等等。如果需要的话,可以将单个图形显示 76 分层以便包含一个或更多视图,例如配置视图、操作员视图、维护视图和仿真视图。可替代地,可以对单独图形显示 76 进行配置,以便利用相同或类似的图形元素 81 来提供这些独立视图,从而在为这些不同功能创建的全部显示提供一致的观感。

[0060] 如框 95 所示,为了向左转到运行时环境 72,可以对图形元素 74 或图形显示 76 进行复制或实例化,并加载到运行时机器上。一般而言,人们希望只有当得到调用或者在运行时机器上实际执行时,才将显示对象 74 或 76 绑定到运行时环境 72 上,此处这称作运行时绑定。也就是说,只有当显示对象实际运行或者在运行时计算机上执行时,才将用于各个实例化对象的解析表填充或绑定到该运行时环境上。该程序确保了当对象的显像在显示屏幕上实际再现时,包含其显像、控件、脚本等的显示对象才执行,且由此使用处理能力。因此,优选是当显示对象在运行时计算机上实际运行时,将该显示对象仅绑定到运行时环境 72 上,这意味着显示对象 74 和 76 可以按照查看这些对象所创建显像的用户的活动定义的方式,间歇地连接至运行时环境 72。特别是,当执行对象需要进行查看时,可以将这些对象绑定到运行时环境 72 上,并且当用户不查看时,例如当用户最小化或关闭这些对象在其中提供显像的屏幕时,可以不绑定或释放执行这些对象。

[0061] 因此,显示对象 74 和 76 是在可以独立环境即配置环境 70 中创建的对象,但是这些对象可以与加工厂环境内定义的其他对象或数据结构捆绑或连接,或者与加工厂环境内运行的任何应用程序捆绑或连接,上述对象、数据结构和应用程序包括例如在任何控制、仿真、维护、或配置环境中定义的对象、数据结构、应用程序等。此外,一旦创建显示对象 74 和 76,它们可以经由解析表定义的直接引用、变量或标签,直接绑定到物理或逻辑过程实体上,或者通过使用别名、变量和参数,间接绑定到物理或逻辑过程实体上,当显示对象 74 或 76 在运行时环境 72 内下载或实例化时,或者在某些情况下,当显示对象 74 或 76 在运行时环境 72 内实际运行时,可以对所述别名、变量和参数进行解析。

[0062] 图 3 的显示编辑器 50 能够在各个细节等级创建显示对象 74 和 76,以便增强显示对象 74 和 76 的使用方便度和多功能性。例如,可以首先创建图形元素 74 以定义更基本的物理和逻辑实体的属性和操作,然后可以通过互连一个或更多图形元素 74 来创建图形显示 76,以便创建更高级或更复杂的显示,这些显示用于描绘更复杂的物理或逻辑实体,或者物理或逻辑实体的群组。当然,图形元素 74 和图形显示 76 两者都可以存储成各种不同类别,并且以各种不同类别进行访问,从而使创建更高级的显示对象对于用户来说更加简单。

[0063] 图4示出了可以由显示编辑器50创建的示例性屏幕显示100。在创建泵的元素的过程中描绘的屏幕100,包括主编辑部分102,元素分级结构部分104,属性定义部分106和显像部分108。主编辑部分102为用户或设计者提供工作空间,以定义或创建该图形元素的显像,并由此定义该图形元素的可视属性,在此例子中,采用显像109进行图示。一般而言,元素分级结构部分104利用分级视图或树结构,提供与主编辑部分102内显像109相关联的部件。在图4的例子中,分级结构部分104示出了在主编辑部分102中定义的显像109包括圆(命名为Circle1)和两个矩形(命名为Rect1和Rect2)的图元或子元素(subelement)。尽管在图4中未示出,分级结构部分104可以包括动画、动作和其他显示特征的指示,例如为显像109定义脚本、可视触发器等。

[0064] 属性定义部分106图示了所有的属性,包括当前为与编辑部分102中所示显像109相关联的图形元素定义的固有属性。图4的示例屏幕100图示了两个属性,包括IsOn属性,它定义了与显像109相关联的泵是开启还是关闭,以及Speed(速度)属性,它定义了与显像109相关联的泵的速度。用户或设计者可以通过在属性定义部分106内定义其他变量、属性等的名称、类型和绑定,将其他属性和参数添加到图形元素上,以便由此定义该图形元素的其他方面。将图4的属性定义部分106中所示的两个属性列为布尔(Boolean)和浮点变量。然而,也可以改为使用其他类型的变量,或者同样还使用其他类型的变量。因此,举例来说,属性定义部分106中定义的属性可以是阵列、表格、枚举列表或任何其他类型的变量或数据结构。

[0065] 如果需要的话,屏幕100内定义的图形元素可以具有与之相关联的多个显像。不同的这些显像可以在显像部分108中进行描绘,并且可以单独进行选择以便置入主编辑部分102中。例如,在图4的显像部分108中示出了两个显像110A和110B,但是可以为所创建的图形元素定义任何其他数目和类型的显像。在图形元素创建过程中,每个显像均可以放置入主编辑部分102中,例如通过在显像部分108的显像指示上右击或双击,将该显像指示拖放到编辑部分102上,等等。一旦放置入主编辑部分102中,就可以对显像进行编辑,以便定义或重新定义其显示属性。一般而言,能够将这些显像中的一个设定或定义为所创建图形元素的默认显像。该默认设定可以以某种方式来表明,例如对图4中显像110A所示的那样,通过高亮显示该默认显像,即在显像部108中用虚线围绕该默认显像,或者以任何其他方式表明。

[0066] 一般而言,由元素编辑器50创建的显像可以由按照所定义的方式放置在一起或者集中在一起的各种形状的一个或更多组件构成。因此,这些显像可以称作形状组件。例如,形状组件可以包括圆、线、点、多边形、正方形、矩形、三角形、或其他简单的图形形状。当以这种方式进行定义时,单独各个动作或动画可以应用于构成形状组件的各个不同形状,或者与这些不同的形状相关联。当然,形状组件可以包括元素的更加精心制作的艺术再现。为了定义或构造形状组件,用户或设计者可以将任意数目的基本图形元素添加到主编辑部分102中,并以任何期望的方式将这些图元集中到一起。一旦创建形状组件,形状组件就定义了实际对象的显像,该实际对象在运行时间中可能实现为XAML对象,当在运行时环境中使用该图形元素时,该XAML对象将在屏幕或显示器上作为显像来显示。构成形状组件的形状或图元可以在分级结构部分104中图示为复合的分级结构。

[0067] 创建单个图形元素的多个显像能够使不同的显像用于不同的场境或者用于不同



的用户,从而为工厂内不同目的创建的显示可以由相同的图形元素来创建(即由相同的图形类对象来创建),同时显示公用元素的不同显像。例如,不同的显像 110A、110B 等可以在不同的功能场境下,从而当该图形元素用作控制操作员显示的一部分时可以使用第一显像 110A,而当该图形元素用作维护显示的一部分时可以使用第二显像 110B,当在仿真显示中使用该泵元素时可以使用第三显像(未示出)。另一方面,在不同类型的显示设备上,不同的显像可以用于各种显示目的。例如,图形元素的第一显像可能适合在典型计算机或工作站上使用,而可以创建不同的显像以便能够方便地在像 PDA 或电话设备之类的手持式设备上使用,并且可以为大屏幕显示器或多屏幕监视器创建另外的显像。应当理解,可以基于要在其上显示显像的显示器的尺寸,来调整用于图形元素的不同显像,从而可以创建适用于大型屏幕如典型计算机屏幕的第一显像,并创建更适用于明显较小屏幕,如像 PDA 或无线电话设备之类的手持式设备的显示屏幕的第二显像。因此,相同图形元素的不同显像可以用于不同的运行时设备中。

[0068] 另一方面,可以在图形元素开发周期的不同时刻,将不同的显像添加到图形元素上。例如,当可能不适用于绘制或创作合意显像的配置工程师第一次创建该图形元素时,可以创作实际上初步的和基本的第一个基本显像(例如线条画型的制图),并在当时将该显像存储为图形元素的一部分。在稍后的日期或时刻,熟练的制图人或能手可以创建更加精心制作的、艺术上合意的显像,并且可以将该第二显像添加到该图形元素上作为备选显像。如果需要的话,在那时,可以将第二显像设定为默认显像,并且该第二显像可以在由主图形元素或类图形元素实例化的所有运行时图形元素上推广,从而该第二显像在运行时显示中显露,或者可以用于运行时显示中。

[0069] 在另一例子中,可以为相同的图形元素配备不同的显像,以便支持不同的显示主题、图形标准、规范或风格。众所周知,不同的工业通常使用不同的图形规范或图形标准来描绘泵、阀门、传感器和其他加工工厂实体。因此,石油和天然气工业使用与制药工业不同的图形标准。利用元素编辑器 50,有可能为各个图形元素提供不同的显像,以便支持多个图形标准或规范,例如石油和天然气规范和制药规范。以这种方式,可以在所创建的图形显示中使用相同的图形元素来支持不同的显示标准、显示规范或显示方案,例如不同工业中可接受的或有用的显示标准、显示规范或显示主题。更进一步地,在不同显像中可以使用不同图形风格,比如艺术风格,以便使用户能够创建各种艺术型的显示。当然,这些仅仅是为相同图形元素提供多个显像的益处处的几个例子,这些多个显像还有其他用途。

[0070] 如果需要的话,图形元素可以具有一个或更多定义的或与之相关联的图形行为。特别是,当在屏幕上显示该显像时,设计者或创建者可以为该图形元素的各个显像定义动画,例如旋转、线性平移、背景变化、色彩变化、恢复到原有尺寸、色彩梯度动画、不透明度动画、字体特性动画、诸如开始/停止特征的视频和视频特征、两维或三维变化,等等。为了添加这些动态行为,用户可以选择一个图形元素,并选择向该显像添加动画(也称作动画例程)。在该点上,用户可以使用例如利用对话框、表达式编辑器等输入脚本,输入有关所选动画的配置信息,或者定义脚本时,这些脚本可以在分级结构部分 104 的分级结构中显露。基本上,脚本就是在运行时活动期间对该显示元素的显像进行查看或者在屏幕上显示时,能够作为显示元素的一部分来运行或执行的程序或例程。尽管并非必要,这些行为或脚本可以捆绑到为该图形元素定义的一个或更多属性或参数上,并且对

这些属性或参数进行操作。例如,脚本可以与某一图形元素相关联,以便基于在该图形元素的属性部分 106 中定义的一个属性值,改变该图形元素的显像内的颜色。例如,当 `IsOn` 属性(该属性绑定到运行时环境上)为 `True`(真)时,即当泵开启时,泵显像 109 的颜色可以从黄色变为绿色。作为另一个例子,可以为泵元素定义一个脚本,它将该泵的 `Speed`(速度)属性与设定点进行比较,并且如果该 `Speed` 属性值高于一定水平,引发显像 109 内某种类型的图形动画。该动画可以包括,例如将泵变红,旋转该泵,显示该旋转的泵内的马达,使该泵显像抖动或振动,等等。

[0071] 例如为了提供该显像的动画或其他行为而为显像定义的脚本,可以设计为对构成该显像的形状组件内的各个形状或图元进行操作,或者对该形状组件内的多个形状进行操作。由于该脚本可以对绑定到运行时环境中的实际物理单元上的固有属性进行操作或者使用该固有属性,因此这些脚本能够使该显像基于实际的工厂操作,或者基于来自加工厂其他区域的、反映与该图形元素相关联的实际实体的属性的数据,而发生变化。

[0072] 应当理解,不同类型的图形元素可以具有可能适用于它的不同图形。因此,可以为图形元素提供的行为不受此处所提供例子的限制。例如,用于旋转设备的图形元素可能包括提供振动图形、运动、颜色变化等的脚本,而用于诸如传感器的设备的图形元素可以包括描述高于或低于界限条件、需要进行校准条件等的脚本。当然,可以为图形元素使用或定义任意属性,并且这些属性一般都基于所表现实体的类型。还应当理解,可以为各个不同显像提供各种脚本,以便为不同显像提供不同的行为。另一方面,某些脚本可以用来为与该图形元素相关联的各个显像提供行为。

[0073] 另外,脚本或其他例程可以与图形元素相关联,以便基于该图形元素的一个或更多参数,检测有关该相关物理实体的特定情况。这些情况可以包括实体的检测状态,包括与该实体相关联的物理状态、类似通信状况的状况、设备状况、数值状况,等等。利用捆绑到由该脚本产生的状态或值中的动画或其他动作或行为,可以将所有这些检测到的情况或状态反映在图形上。例如,可以为泵图形元素提供这样的脚本:基于该元素的一个或更多参数,也就是在图 4 的属性部分 106 中定义或披露的一个或更多参数,检测相关泵的过热状态。作为另一个例子,可以为图形元素提供一个脚本,以便检测该泵的过度振动、或任何其他情况。如果检测到诸如过热或过度振动的情况,则与该图形元素相关联的动画或其他动作可以针对该情况进行操作,以便在该图形元素的显像内或显像上提供该情况的图形指示。

[0074] 图 5 图示了可以由显示编辑器 50 生成的另一示例性屏幕显示 112。与图 4 的屏幕 100 类似,该屏幕显示 112 包括主编辑部分 114,制模板(pallet)视图 116,分级视图 118 和属性视图 120。在主编辑部分 114 中描绘了反应器元素 122 的第一显像。如分级视图 118 所示,该元素的标题为 `Reactor1`,并且该元素包括命名为 `Visual1`(默认显像)、`Visual2` 和 `Visual3` 的三个显像。正如在分级视图 118 中标题 `Visual1` 的下面所示,第一显像由包括矩形元素和椭圆元素的 `Canvas`(画布)背景组成。在属性视图 120 中列出了当前定义的属性,在这个例子中是显像的名称、高度和宽度。当在分级视图 118 中选择显像时,在编辑视图 114 中展现与该显像相关联的任意图元或元素,并且在属性视图 120 中显示当前所选元素的属性。

[0075] 在屏幕 112 中,制模板视图 116 包括可以用来创建显像的若干基本元素。例如,制

模板视图 116 包括一组基本 UI (用户界面) 元素,例如按钮、文本框、滑动块、旋纽等,一组基本面板,以及一组基本形状。所定义的面板可以包括画布 (canvas) 面板,它定义了这样一个区域,其中用户可以通过相对于画布区域的坐标明确地定位各个元素;还可以包括平台 (dock) 面板,它定义了这样一个区域,其中用户可以相对于彼此水平或竖直地排列各个元素;以及包括流动 (flow) 面板,它可以用来沿着所指示的流动方向,在流动面板区域内断开、弯曲和对准其内容。所述流动方向例如可以是上、右、左和下的任意组合,例如从左到右从上到下,或者从右到左,从上到下,等等。更进一步地,制模板视图 116 中的基本形状可以包括 ISA (美国仪表协会) 符号、变送器符号、阀门符号、PI&D 图符号或其他控件符号等,或者任何其他期望的形状,所有这些都可以用来建立图形元素。

[0076] 当然,其他基本元素,例如基本控制元素、设备等也可以配备为制模板视图 116 中的图元,以便用来创建所定义图形元素的显像。制模板视图 116 还可以提供用户定义类别或元素的一个列表,可以创建该列表以便允许用户将任何其他有用的形状从制模板视图 116 拖拽到编辑视图 114。如果需要的话,可以将对这些用户所定义类别的访问限制为创建它们的用户,并且这些类别和元素可以存储在与特定用户相关联的用户偏好文件中。然而,内嵌的类别和制模板项目可以存储在数据库中,并且可以全局地用于所有的用户。无论如何,制模板视图 116 可以用来显示或提供对基本元素库的访问,这些基本元素可以用来组成图形元素,并且这些库可以根据需要进行锁定、变型、限于特定的用户,等等。

[0077] 如上所述,用于图形元素的任何显像都可以具有与之相关联的动画和/或动作,并且这些动画或动作可以在屏幕 112 的动作/动画视图 123 中进行展示。当显像包括动画或动作时,这些动画或动作还可以用专用符号如星号在分级结构中表明。当在分级视图 118 中进行选择时,将在动作/动画视图 123 中显示为显像或显像子元素定义的任何动作或动画。可以通过在视图 123 中定义这些动作或动画,或者通过将这些动作或动画添加到分级视图 118 中,来创建和分配动作或动画。当用户希望创建或编辑一个动作或动画时,编辑器 50 可以提供一个对话框或编辑框,以便允许充分地规定或定义该特征。当然,可以利用脚本、可视触发器或其他程序来定义动作或动画。

[0078] 在屏幕 112 的使用期间,用户或设计者可以通过拖放或其他方式选择制模板视图 116 中的不同项目,并将这些项目排列在编辑视图 114 中以创建期望的显像,从而创建一个元素的显像。一个或多个工具栏 124 可以用来提供任何标准的编辑功能,例如添加新的显像或动画,删除、移动、编辑动画,例如从前向后排列图元或元素,提供连接元素以便显示或实现对所创建显像要进行的连接,将不同的图元集中到一起以便它们保持它们相对于彼此的位置,添加像线条和文本的静态元素,等等。

[0079] 如图 6 所示,图形元素 130 可以具有与其相关联的多个显像 132 或可视表示,可以理解在不同时刻,或者在使用图形元素 130 的不同显示中,可以使用不同的显像 132。如上所说明的,显像 132 中的任何一个都可以由任意数目的图元 134 以及动画和动作 136 组成。更进一步,图形元素 130 可以包括任意数目的属性或参数 138,这些属性和参数可以被捆绑或者用于动作和动画 136 中,以便对显像 132 进行改变,以及操作与显像 132 相关联的可视触发器 (visual trigger)。更进一步,显像 132 或组成显像 132 的各个图元可以具有为预定事件定义的动作,例如鼠标跳过 (over) 事件、鼠标点击事件等。这些动作 (也称作例程) 能够对事件处理器进行设定或定义,以便进一步定制图形元素 130 的状态,并允许用户

与显像 132 交互,从而例如引起运行时环境内的变化。特别是,用户可以通过在其中输入数值或者其他数字或信息、移动显像上的元素如滑动条、或者通过采取用来改变例如显像内属性的其他动作,来与显像 132 交互。该显像属性变化可以经由脚本或者直接捆绑到过程输入,例如过程运行时变量上,以便使得该变量发生变化。特别是,显像属性可以被连接至为图形元素定义的属性上,而图形元素进而又可以被绑定到过程输入上。采用这种方式,用户可以经由动作或动画例程与显像进行交互,从而引起变化或者向诸如仿真环境之类的过程或其他运行时环境提供输入。

[0080] 如果需要的话,可以通过由用户指定或者通过编程方式指定,来将显像 132 连接至图形元素 130。具体地说,图形元素 130 可以显露出枚举属性,该枚举属性允许基于例如由该图形元素代表的元素的制造商,或者与该图形元素 130 相关联的其他参数,例如与该图形元素相关联的设备状态,来以编程方式改变该显像。

[0081] 如上所述,图 5 的属性视图 120 提供或示出了为分级视图 118 中的所选项目,并由此为主编辑视图 114 中所描述项目定义的属性和事件。用户可以利用例如工具栏按钮,在属性视图 120 内的属性、固有属性和事件当中来回切换。图 7 示出了属性视图 120A,其中示出了固有属性,在这种情况下包含 IsOn 和 Speed 属性。这样,属性视图 120A 示出了这些变量的数据类型和这些变量的任何默认设置。如上所述,用户可以向其上添加、从其上删除或者编辑该属性列表,以便定义具有任何期望的数据类型的属性,包括枚举、表结构等。如果需要的话,属性视图 120 还可以显示为这些属性定义的任何运行时绑定,并且这些运行时绑定可以是固定变量或标签,或者可以是使用在运行时间填入或者在向运行时机器下载该图形元素时填入的别名的标签。

[0082] 为图形元素定义的属性值可以被用作动画、动作等的触发条件,并且这些状态可以由与该图形元素相关联的一个或更多个脚本来定义。例如,如果马达的开启 (On) 属性为真,则可以在图形显示中触发马达运行的动画。更进一步,图元属性,例如矩形形式的图元的矩形填充属性,可以被绑定到图形元素属性上,从而图形元素属性的变化会影响图元属性。同样,显像变化可以被捆绑到属性上,从而显像变化可以引起属性值的变化。

[0083] 如果需要的话,编辑器 50 可以使用户能够规定一个或更多变换函数 (transformer function),以便在图形元素属性与动画或动作之间提供更合乎需要的绑定。例如,用户可能希望将 TankLevel (定义罐体内的液位) 的图形元素属性绑定到图元属性矩形填充上,自此通过被定义为该显像一部分的图元的填充颜色来图解说明罐体液位。然而,在这种情况下,用户可以定义将该属性 (TankLevel) 变换为枚举型集合或条件的变换函数,从而如果罐体液位在第一和第二液位之间,将矩形填充设置为绿色,如果罐体液位在第二和第三液位之间,将矩形填充设置为黄色,如果罐体液位高于第三液位,则将矩形填充设置为红色。该变换函数可以被定义为利用图形元素来执行的脚本或任何其他程序,并且可以用来产生任何期望的属性变换,例如将属性值变为长度、字体名称、定位串 (localized string)、持续时间、旋转、颜色梯度、不透明性、画笔图案,等等。同样,变换例程可以通过显像来自用户的输入变换为任何期望的属性值。

[0084] 作为另一个例子,可以利用变换函数将图形元素属性或参数绑定到外部引用上。此处,编辑器 50 可以允许用户规定一个或更多个变换函数,当图形元素在运行时环境中使用时该变换函数能够被自动加载,以将源值 (运行时变量) 转换成宿值 (图形变量),反之

亦然。例如,可能被绑定到提供速度数值的数据源的速度宿变量,可以基于源变量的值,转换成字符串型变量,例如表示“慢”、“中等”和“快”的字符串。这些变换函数还可以用来转换单位,或者提供其他变换。无论如何,这些变换函数可以作为脚本或其他程序来执行,并且可以用于任意目的以提供动作或动画,或者以其他方式影响图形元素的显示属性。

[0085] 当然,可以为过程控制内不同物理单元创建的不同类型的图形元素定义任何期望的属性。例如,泵元素可以包括泵状态(例如,开或关)、入压力属性、出压力属性、入流属性和出流属性。同样,关于致动器使用的调节值可以包括,例如名称属性、入密度属性、出密度属性、入流属性、出流属性、入压力属性、出压力属性、入温度属性、出温度属性、阀门位置属性、阀门-开和阀门-关属性(可以定义阀门是一直开启还是关闭)、设定点、过程值、诸如线性、快速开启、等百分比之类的阀门类型,等等。当然,该列表并非意在全面详尽。更进一步,这些属性中的任何一个都可以被连接至图形元素的动画或动作。

[0086] 图8描绘了第二属性视图120B,它可以用来查看与分级视图118(图5)内所选元素(在这种情况下一般为Reactor1元素)的图形元素属性相关联的动画和绑定。图8的属性视图120B示出IsOn属性与该图形元素的画面中各个部分相联系的方式。特别地,如表项目140所示,存在与显像背景相关的、并且基于IsOn属性值操作的动画。表项目142示出IsOn属性之间的绑定,并且可以用来访问将该IsOn属性联系到该显像内背景元素或图元的动画、脚本和变换。在这种情况下,用户可以通过选择表项目框140和142中的按钮,获得额外的信息。例如,通过选择表项目140中的按钮,用户可以访问使显像背景基于IsOn参数值改变颜色的动画。点击这样的动画按钮还可以使编辑器50打开允许对动画进行管理和配置的属性动画对话框。更进一步,通过点击框142中的绑定按钮,可以执行和管理绑定,这可以使编辑器50打开能够对绑定进行创建、查看和编辑的对话框。当然,也可以为图形元素的其他属性配备类似的屏幕,而在属性屏幕120B中显示的属性列表将取决于分级视图118中当前所选的项目。

[0087] 再次参见图5,当在分级视图118中选择显像时,属性视图120将显示该显像属性,例如名称、提供该显像唯一标识符的I.D,以及该显像是否被设置为该图形元素的默认显像的描述。分级视图118或属性视图120还可以表示该显像是否被锁定,即它是否可以进行修改。

[0088] 另外,当从分级部分118中选择连接器元素时,属性视图120将显示该连接器元素的属性,它可以包括连接器类型(例如,液体管线、电线、气体管线、圆管或方管、传送带等)的指示,该连接是进入该设备还是从该设备中出来(即相对于该图形元素所表示的实体的物质流方向),所需或所允许连接的最小和最大数目,该连接的宽度和高度,该连接相对于图形元素的位置,例如顶部、左部等。

[0089] 当在分级视图118中选择图元时,属性视图120将显示该图元的属性,可以包括例如I.D,该图元是可选的和/或可视的类别描述,该图元形状或特性的尺寸和长度或其他定义,该图元的背景颜色和填充空间,等等。当然,分级视图118中任何其他所选的项目也能够使该项目的属性被显示于属性视图120中,并且该属性的类型和特性取决于所选项目的特性。

[0090] 如果需要的话,可以提供事件板以显示与分级视图118内各个或全部所选分级项目相关联的事件。这种事件板可以包括当用户执行有关图形元素的动作时发生的事件,例

如“点击”事件和“鼠标跳过事件”，这些事件定义了当用户在图形元素或其一部分的显像上点击时所发生的事件，或者当用户在图形元素或其一部分的显像上定位鼠标指针时所发生的事件。作为该事件板的一部分，可以向用户提供对脚本编辑器的访问，以便定义或访问能够在事件发生时运行的脚本（此处这被定义为例程）。

[0091] 另外，如果需要的话，图形元素可以具有与之相关联的定制事件。定制事件通常是被定义为某些外部或外界事件结果的事件，或者是由于图形元素的用户所采取的行动，需要与外界应用程序或数据源进行通信的事件。实质上，图形事件是由图形元素发送的消息，用来发送出现了关于该图形元素显像的运动的信号。所包含的应用程序或者执行图形元素的运行时应用程序可以利用如 C# 语法，或通过以任何已知的方式注册到事件通知，来捕获这些通知。在一个例子中，所包含的应用程序可以注册到泵过热定制事件上，并提供能够运行脚本或其他例程的“我的处理器”功能，以使用户在事件触发时能够处理该事件。当图形元素作为控制操作员屏幕的一部分实施时这些定制事件尤其有用。在这种情况下，例如通过将泵热量属性（被绑定到外部测量上）与设定点或其他界限进行比较，图形元素自身就可以确定事件状态，并且可以通过向用户通知该事件并且执行应用程序、脚本、对话等来触发事件响应，从而使用户能够处理或处置该事件，这里该事件是过热泵。为了实现该事件的处理，元素编辑器 50 能够允许用户定义事件自变量，即哪些被认可为事件，以及哪些是由发生该事件而引起的。

[0092] 当然，如果需要的话，图形元素可以提供从 Avalon 控制类继承下来的标准事件。众所周知，Avalon 对象或控件是 Microsoft Longhorn 操作系统这对 Avalon 用户界面架构的 Microsoft 定义的图形，它支持向量图形在用户界面上的再现。这些标准事件可以包括，例如将对面板显示或设备细节显示的查看捆绑到点击或双击（鼠标）事件上。例如，点击或双击事件，作为报告图形元素显像的绑定内鼠标点击或双击的事件，可以触发对该显像所代表的设备或实体的面板显示或细节显示，并且向用户提供有关所代表实体的状态、设计、制造等的更全面或更详尽的信息。当然，可以为作为报告按键受到按压事件的按键下 / 上事件，为包括鼠标输入、鼠标悬停 (hover)、鼠标移动、鼠标滚动等动作的鼠标事件，或为任何其他用户发起的事件，定义包含标准事件在内的其他事件。

[0093] 因此，如果需要的话，可以根据图形元素属性内的变化，在事件处理器的脚本内，例如在用于基本事件的事件处理器中，或者在变换函数内，触发图形元素事件。更进一步，图形元素的图元可以显露出能够通过与它们进行用户交互来触发的事件，例如通过鼠标和键盘与用户交互。这些事件为用户提供选项，以在形状和图元级与图形元素进行交互。举例来说，开发者可以通过指定利用 C# 方法的事件处理器，在图形元素的内部处理这些事件。

[0094] 再次参见图 5，动作 / 动画视图 123 可以提供或示出为当前显示于编辑视图 114 中的显像而定义的变换动画和属性动画的列表。例如，用户可以双击视图 123 内的行，以使编辑器 50 显示对话框，该对话框可能是变换动画对话框或属性动画对话框，它允许用户编辑所选择的动画。通常，仅显示当前所选显像的动画，但是选择全选 (Show All) 框 144 可以显示所有显像的动画和动作。假设这些动画具有富有意义的名称或描述，例如“激励 (animate) 杠杆”、“旋转马达”等，动作 / 动画视图 123 就可以针对显像定义的动画状态提供容易的查看和访问。图 9 示出示例性动作 / 动画视图 123A，该视图示出了为 IsOn 属性定义的动作。在这种情况下，当 IsOn 属性为“真 (True)”时，被称为“Visual1”的显像的

“Rectangle1”图元执行以填充值“红色”进行填充的填充动作。同样,当 IsOn 属性为“True”时,“Visual1”显像的“Ellipse1”图元执行带有错误值的可见动作(例如,变为不可见)。当然,可以为显像的各个图元分别定义动作和动画,但是这些动作和动画可以基于相同的事件、触发或属性变化而同时操作,以便显示更复杂的但合意的动画。更进一步地,动作和动画可以包括执行单次操作,例如增大尺寸、填充颜色等,或者执行在关闭之前连续发生的重复操作。

[0095] 图 10 示出了基于诸如固有属性之类的图形元素 152 的属性,可视触发器 148 (visual trigger) 可以在图形元素 152 的显像 150 上实施的方式。具体地说,由可视触发器 148 定义的一个或多个属性触发器 154 (在图 10 中示出了其中的三个) 来监控图形元素 152 的一个或多个固有属性。可以利用脚本来实施的各个属性触发器,如虚线 155 所示可以监控某个图形元素属性值。其后,如虚线 156 所示,当所监控的图形元素属性值满足或符合规定条件时,各个属性触发器可以设置该显像 150 的一个或多个图元属性值。因此,举例来说,一个属性触发器 154 可以监控一个或多个图形元素属性,以便确定一个或多个这些属性的值何时落在特定范围之内。当符合该条件时,一个属性触发器 154 可以使动画或其他可视化脚本在例如显像 150 的图元或其他元素上运行,以便提供可视触发器 148。当然,多个属性触发器 154 可以共同操作,以便提供作为可视触发器 148 一部分的多个同时发生的变化或动画,或者不同的属性触发器 154 可以独立地对不同的固有属性进行操作,或者基于相同固有属性的不同值独立地操作,以便在不同时刻或者响应于不同的过程条件提供可视触发器 148 的不同操作。以这种方式,基于图形元素 152 的固有属性值,可以在显像 150 中提供颜色变化、动画等。

[0096] 如果需要的话,编辑器 50 可以提供或显示可视触发器面板,该可视触发器面板列出了当前所选显像的所有属性触发器。在图 9 中示出了这样的面板,其中元素属性列中列出了图形元素属性名称,第一数值列表表示所观看的图形元素属性值,目标列提供要发生变化的显像或图元标识,路径列是变化的图元属性,而第二数值列是当满足所观看的图形元素显像值时要应用的图元属性。当然,可以利用该结构来提供动画和其他显像变化。

[0097] 图 11 示出变换动画对话框 160,它可以用来帮助用户规定变化动画,或者使用户能够规定变换动画。如图所示,变换动画对话框 160 包括移动 (move) 部分 162,它允许用户依据定义动画内运动的像素来规定方向和距离,还包括旋转 (rotate) 部分 164,它允许用户规定动画的旋转方向和角度,以及包括缩放 (scale) 部分 166,它允许用户为该动画规定水平和垂直方向上的缩放度,并且规定是否要锁定纵横比。该对话框 160 还包括偏斜 (skew) 部分 168,它允许用户规定在动画期间,在水平方向和垂直方向上要施加的偏斜。设置 (settings) 部分 170 允许用户定义该动画是否是连续的,并且能够实现其他移动、旋转、缩放和偏斜动作中的任一个。更进一步,预览框 (preview) 172 可以图示该动画的预览。

[0098] 应当理解,动画是值随着一段时期发生变化的对象。通过关联动画与图元属性可以获得属性动画。属性动画可以尽可能精细,从而使文本颜色发生变化,或者可以使像线一样的元素发生闪烁。另一方面,属性动画可以更加复杂,例如制作折线中各个点的动画,等等。当然,这些仅仅是动画的一些例子,而其他动画可以包括添加或改变颜色,改变图元的尺寸(例如宽度、长度或点尺寸)、移动、旋转、偏斜、缩放图元,等等。更进一步,在显像的任意级,例如在图元级或者作为整个显像的一部分,都可以提供其他动画。当然,如果需要

的话,多个动画可以被提供给或应用于任何特定的显像,或者被提供给或应用于显像中任何特定的图元。更进一步地,基于属性值的变化或者基于用户发起的触发事件,例如鼠标事件,与特定显像或显像的图元相关联的多个动画可以同时操作或者在不同时刻操作。

[0099] 图 12 示出了属性动画对话框 180,它可以由编辑器 50 产生,以使用户能够定义或改变动画的属性,由此定义该动画。对话框 180 包括绑定定义 (bounds) 部分 182 和时间线 (timeline) 定义部分 184。绑定定义部分 182 提供或定义动画的绑定。具体地,“从”属性定义动画的开始值,而“到”属性定义动画的终止值。时间线定义部分 184 定义动画的持续时间、动画的开始时间和终止时间。持续时间属性定义动画从开始到完成的时间长度,开始时间属性定义相对于动画开始时间的时间偏移,而终止时间属性定义相对于起始时间的动画终止时间。速度 (speed) 定义部分 186 允许用户利用例如滑动条等,规定速度、加速度和减速度。显而易见,速度属性定义动画的速度,加速度属性使得动画相对于时间增长移动得更快,而减速度属性使得动画相对于时间增长移动得更慢。重复 (repeat) 定义部分 188 允许用户定义一种方式,在该方式下,举例来说,动画能够重复规定量的时间,重复规定的计数或持续时间,或者在关闭之前一直连续地重复。

[0100] 可以通过对话框 180 进行访问的另一动画对话框 190,可以用来指定不同的颜色,以便当使用绑定定义部分 182 时,用作动画中的绑定。类似的对话框可以用来使用户能够将其非数字值选择为动画的属性。尽管未被示出,除了利用该对话框或其他对话框以外,也可以选择或规定其他的动画属性。例如,自动反转属性可以用来定义布尔 (Boolean) 值,它表示动画在完成其前向播放之后是否反过来播放,而“by”属性可以用来定义动画改变其起始值的总量。相对速度属性可以用来定义与父动画相比,即与该显像的父元素相关联的动画相比,时间流过该动画的相对速度。例如,值“1”可以表示该动画的进行速度与父动画相同,而值“2”可以表示该动画的进行速度是其父动画的两倍,等等。

[0101] 尽管未被示出,对话框还可以用来设置动画的字体 (font) 属性,例如样式、尺寸、字形,以及动画内的文本颜色或其变化。另外,笔画属性对话框可以用来设置边沿或线型、线的粗细和颜色,以及这些属性的变化。

[0102] 图形对话框还可以用来编辑图形元素的其他特征。例如,图形对话框可以用来创建新的过程图形,包括将图形元素属性添加到图形元素上,并将图形元素属性绑定到运行环境上。这样的绑定对话框可以提供一种浏览器,该浏览器使用户能够浏览控制系统或其他运行时环境内定义的不同标签或变量,以便对所需的标签、变量、名称等进行定位,从而执行绑定。同样,其他对话框也可以用来添加显像、添加可视触发器、浏览图形元素或显示以及添加事件。

[0103] 因此,应当理解,图形编辑器 50 为定义图形元素固有属性提供支持,为图形元素显像提供支持,包括创建和操纵图形图元或形状,添加动态行为、特定变换动画 (旋转、平移、缩放和偏斜) 和属性动画 (长度动画、颜色动画等),以及定义触发动态行为执行的条件。更进一步地,图形编辑器 50 为在数据库中存储和检索图形元素提供支持,包括对于将图形元素串行化为 xml blob 的支持,并且为在图形元素库中对图形元素进行分类提供支持。此外,该系统还提供以下功能:将图形元素存储成用户定义类别或集合,锁定或提供这些图形元素的安全性,将图形元素存储成用户定义类别或者其他有用的类别,对图形元素进行版本化 (version),等等。



[0104] 无论如何,一旦创建了图形元素,图形元素就被存储在图形元素数据库中,使得它并不被绑定到加工厂运行时环境内的过程变量或其他数据上。该被存储的图形元素并不必用于任何显示中,但是可用于如下使用:该元素现在能够被下载至运行时环境,并且能够被绑定到加工厂或过程控制系统内特定的实际或仿真的物理单元上。当图形元素被绑定时,将图形元素的固有属性,例如过程变量、设定点、当前速度等捆绑和绑定到运行时环境内的数据引用上。

[0105] 图 13 示出了可将图形元素捆绑或绑定到加工厂内或者在加工厂中使用的过程控制系统内的实际物理部件或元件的一种方式。具体地说,各个图形元素 192 包括实质上定义元素显像的 XAML 194,以及相关的脚本 196、动画 198、触发器 200、动作 202 和事件 204。图形元素 192 还包括引用表 (reference table) 206,它列出或包括了与 XAML 194 相关联的、在 XAML 194 中可用的或者开放给公开的所有引用。引用表 206 实际上由图形元素的属性和参数组成,或者包括图形元素的属性和参数,以及脚本 196、动画 198、触发器 200、动作 202 和事件 204 所使用的任何其他变量或引用。引用表 206 内的变量或实体可以引用或者被绑定到其他程序中的变量、表格、表格项目,或者引用或者被绑定到过程控制系统中其他地方定义的任何其他类型或种类的数据。

[0106] 如图 13 所示,解析表 208 用来将引用表 206 内的引用或变量捆绑到实际过程控制环境或其他运行时环境。一般而言,当图形元素 192 实际上被配置为用于运行时环境内的特定显示,并且下载至运行时机器时,为图形元素 192 提供或创建解析表 208,它可以直接地或经由别名化来定义这些引用。在对显示内的图形元素 192 进行操作之前或操作期间,解析表 208 解析这些别名和其他参数,并且将这些解析出的数据连接捆绑到引用表 206 上,以便在引用表 206 内的变量与过程控制系统或其他运行时环境内的实际数据源之间提供绑定。

[0107] 为了避免在运行时间期间不必要地使用过程控制系统内的处理能力,当不在任何显示屏上实际显示或使用图形元素和使用这些图形元素的图形显示时,它们不需要保持被绑定到运行时环境上。相反,只有当图形元素 192 在运行时环境内的屏幕上运行或进行显示时,才可以将解析表 208 绑定到图形元素 192 的引用表 206 上。

[0108] 由于通过使用引用表 206 和解析表 208 将各个图形元素捆绑到过程控制系统或时间运行环境中,并且由于绑定发生于在系统级创建图形元素、并进行复制且装载到运行时环境或机器中之后,因此图形元素 192 可以在不同的时刻,为了不同的用途而分别被绑定到不同的数据源。更进一步,图形元素 192 可以被绑定到由任意数目的不同源生成的或者通过任意数目的不同源可以得到的数据上,包括图 2 的不同功能区域 42 内的数据源,例如与控制活动、维护活动、建模活动、仿真活动、配置活动等相关联的数据源。

[0109] 例如,如图 14 所示,可以将显示元素 192 捆绑到,并且可以用来显示或处理在极其不同类型的数据源中生成的,或者可以从极其不同类型的数据源中可以得到的数据。这些数据源可以包括控制环境数据源 210,像诸如众所周知的 DeltaV 控制系统之类的控制器程序,还包括 OPC 数据源 212,它通过众所周知的 OPC 连接接口提供到其他系统的连接,还包括像众所周知的 AMS 系统的维护数据源 214,像众所周知的 Ovation 系统的更高级的系统或商业系统 216,甚至还包括像竞争性控制应用程序之类的使用竞争性系统 218 的数据源。以这种方式,即使数据来自或源自极其不同类型的应用程序,包括与永不意在共同操作的

竞争性系统相关联的应用程序,通过利用来自该系统内任何应用程序的数据,图形元素 192 也可以用来在系统的任何级别显示物理单元的数据或显像。因此,由于在系统级创建图形元素和从图形元素创建的图形显示,因此即使不同类型的软件正在访问和运行该工厂内不同的硬件和软件部件,也可以出于任何目的将它们用来提供该工厂内所发生一切的显像。

[0110] 由于图形元素的模块化特性,有可能通过用户所周知的且被充分支持的设备描述语言 (DDL) 来编写的设备描述 (DD), 自动或半自动地创建图形元素。具体地,设备制造商通常为他们所制造的各个设备提供 DD, 该 DD 用 DDL 定义了与该设备相关联的参数, 如何与该设备进行通信, 对该设备的限制, 等等。因此, 图形元素创建应用程序可以为用 DDL 编写的设备读取 DD, 以便确定设备类型和与该设备相关联的重要参数、限制等, 从而可以将这些参数定义为该设备的图形元素的固有属性或参数。该程序还可以将基本的形状组合选择或定义为该设备的显像, 并且可以选择一个或更多个通用脚本, 以便基于来自该 DD 的信息, 或者基于为该设备的 DD 所定义的设备类型的设备而存储的模板, 来提供该设备的基本动作和动画。如果需要的话, 在该过程期间, 该程序可以询问用户以提供关于设备的信息, 或者对哪些动画、显像、图元等用于该图形元素作出选择。

[0111] 对于更完备的或明确的图形元素, 该程序可以存储用于不同类型的设备的各种通用图形模板, 例如用于传感器、阀门、马达、罐等等。然后该程序可以基于该设备的 DD 所定义的设备类型, 来确定要使用的模板图形元素。如果需要的话, 该模板可以提供或者具有可用于该图形元素的各种选择或选项, 并且这些选择可以基于该设备 DD 内的信息或者基于用户输入来确定。因此, 举例来说, 该模板可以提供与诸如传感器之类的各种设备子类型相关联的各种固有参数, 并且该程序可以基于该 DD 内的信息, 确定用来定义该图形元素的固有参数。

[0112] 各种基本脚本可以被配备为模板的一部分, 并且可以用来提供显像的行为。同样, 可以基于 DD 内的信息, 例如设备类型等, 或者如果需要的话, 可以基于对用户的问题, 来自动地选择要在图形元素中使用的脚本。另外, 可以根据 DD 内的信息, 确定用于该脚本中的各种限制或变量。因此, 举例来说, 如果 DD 表示该设备是一台旋转设备, 那么可以选择诸如提供旋转图形的一个脚本, 以便在图形元素内使用或配备该脚本, 并且当该图形改变颜色等时, 该旋转图形的某些方面, 例如速度, 可以以与该 DD 内定义的设备相关联的限制为基础。这些限制例如可以是, 正常或额定的运转速度, 所定义的超速或速度不足条件或限制, 等等。作为另一个例子, 如果该设备是传感器, 该传感器的高值和低值可以被用来提供与该传感器当前读数相关联的图形, 以及描述该传感器是否损坏的图形, 等等。

[0113] 以这种方式, 当基于该设备的 DD 内定义的设备类型和设备的已知性质, 通过为该图形元素定义某些基本脚本、图形动画、显像和固有参数, 在加工厂内对该设备进行归属或辨认时, 可以根据该设备的 DD 自动地创建基本图形元素。这种图形元素的自动创建向用户提供某种能力, 以将新近添加的设备自动集成到图形中, 如果有的话, 也不必对该设备执行许多图形编辑, 从而当用于这些设备的 DD 被加载到系统中时, 至少在基础级, 可以在图形显示中自动地支持该设备。换句话说, 通过为系统提供设备的 DD, 然后运行该程序以便根据该 DD 创建该设备的图形元素, 用户可以根据设备的 DD 来自动创建图形元素。其后, 该图形元素可以用于一个或更多个图形显示中, 以便建模 (model) 或提供该设备的显像, 这向用户提供这样的能力, 能够对该设备建模或者在图形显示中图示该设备, 而无须手动地创建

该设备的图形元素。

[0114] 如上所述,一旦创建了大量图 3 中的图形元素 74,图形编辑器 50 可以用来创建一个或多个图形显示,例如图 3 中的显示 76。实际上,如果需要的话,可以向用户或购买者提供带有人量预先配置的图形元素的图形显示器 50,这些预先配置的图形元素可以具有与之相关联的各种显像,例如用于不同工业、不同功能用途的显像,等等。从而,图形编辑器 50 可以允许用户创建定制元素,以及创建或建立图形显示 76。

[0115] 一般来说,为了创建显示 76,用户将从图形元素 74 和其他可视元素的库中进行选择,并将它们放置到一起以建立显示。在完成图形显示之后,可以在数据库,例如在配置数据库中,将所得到的数据结构或对象保持为显示类对象,该显示类对象具有所有这些各种元素、固有属性以及被定义为单个定义实体的显像。然而,该显示类对象能够不被绑定到过程变量上,并且不必用于任何运行时显示中。如果需要的话,可以在数据库中将该类对象保持为 XML blob,它具有全部被存储和联接在一起作为 XML 实体的显像、脚本等。其后,可以根据该类对象创建这些单独的图形显示,并且这些单独的图形显示可以被分配给且下载至操作员工作站或其他运行时环境中。

[0116] 当下载显示时,会将其中的图形元素定义转换成 Avalon 控件,编译为汇编程序,并且配置在目标机器即运行时机器上。其自身为 Avalon 实体的显示,会引用所编译的控件汇编程序,另外,可以生成定制数据源,它充当将诸如控制或维护运行时应用程序之类的 Avalon 控件连接到后端数据源的数据适配器。以这种方式,可以用一种语言来创建或编辑图形显示(如果需要的话,可以创建或编辑各个图形元素),并将其存储为另一种语言或形式(例如 XML blob),并且以第三种语言或形式(例如作为被捆绑到用各种可执行语言编写的脚本上的 Avalon 控件)来运行。

[0117] 现在参见图 15,屏幕 220 可以由图形编辑器 50 来产生,以便使用户能够创建一个或多个图形显示。一般而言,图形显示由代表工厂内物理设备的相互联系的元素组成,并且可以包括或者显示与这些设备相关联的附加信息。图形显示中的每个图形元素均包括一组等同于过程变量、恒量、或者其他外部值的固有属性,并且如上面所说明的,每个图形元素均可以具有若干可视表示,它可以包括动态行为,例如颜色变化或动画。另外,该图形显示可以包括静态元素,例如文本、框等,允许用户以某种方式与该图形显示进行交互的用户动态图标,以及可以向用户显示过程或其他信息的变量框,等等。

[0118] 与用于图形元素的图 5 中的编辑屏幕 112 类似,图 15 的示例性图形显示编辑器屏幕 220 包括在其中建立图形显示的主编辑画布 224,并且还包括选项板部分 226,其中可以显示模板图形元素、图元或其他库元素,并且通过该选项板部分可以选择这些不同的元素,并将其拖放到主编辑画布 224 上。主编辑画布 224 提供图解视图,其中用户交互以创建和编辑图形,或者是图元或组件,并且提供向量图形编辑和查看特征。编辑器 50 使用户能够将图形对象,例如图形元素,放置到无限空间中的任何地方,它可以在编辑画布 224 中当前所显示的显示帧的内部,也可以在其外部。因此,主编辑画布 224 可以只描绘当前所创建显示的一部分,并且用户可以扫描(pan)该视图以便显示该显示的其他部分,放大或缩小以改变该视图的放大程度,以及旋转该视图以获取最便利的方向,以便在任何给定的时刻都能够在该方向上对显示进行作业。然而,应当理解,在主编辑画布 224 中可以完成图形对象的所有绘制和操控。为了帮助绘制,可以根据标尺 227 读取视图的 X-Y 位置,可以在工具栏

228 上的下拉列表中显示放大程度,并且可以通过选择缩放弹出菜单上的表项来改变放大程度,等等。

[0119] 屏幕 220 还包括分级部分 230,它描述主编辑画布 224 内元素的分级显示或列表,还包括属性部分 232,它列出与画布 224 中所创建的显示或者画布 224 内高亮元素相关联的属性或参数,并且包括绑定部分 234,它图示或列出了这样一种方式,通过该方式将不同的参数或特征绑定到过程控制系统或任何其他运行时环境内的元素上。另外,屏幕 220 包括工具栏 236,它列出与在主编辑画布 224 中所创建的图形显示相关联的各种视图或者图层。更具体而言,任何特定的显示都能够具有可以由不同的人在不同的环境下使用的各种图层或视图,例如操作员显示、维护显示、仿真显示、建模显示、工程显示、商业显示,等等。在图 15 的示例性屏幕 220 中,工具栏 236 包括三个视图或等级,图示为提供典型的控制操作员视图的操作 (Operation) 视图 (当前在主编辑画布 102 中示出),提供典型的维护或工程师视图的工程 / 维护 (Eng/Maintenance) 视图,以及训练 (Training) 视图,该训练视图提供在对加工厂或者对所创建的显示中描述的加工厂一部分进行仿真时使用的仿真视图。由于没有图 15 的主编辑画布 224 内选择的元素,属性部分 232 显示所创建显示的前名称,连同该显示的特征,例如宽度和高度、创建者提供的描述、名称、背景描述,以及创建者可能希望为该显示存储的任何其他信息。同样,由于没有实际元素被放置于图 15 的主编辑画布 224 中,因此分级部分 230 仅显示该显示的名称,并且在绑定部分 234 中不显示任何绑定。

[0120] 在显示的创建期间,例如通过经由选项板部分 226 内定义的一组元素类别之一来访问该元素,用户可以将诸如图形元素之类的元素放置到该主编辑画布 224 上。在图 15 的示例中,选项板部分 226 示出其可以被放置到主编辑画布 224 中、且被连接到一起以创建完整显示的不同元素类别,包括致动器元素、计算和控制元素、处理元素、属性和测量、形状、用户界面控件和用户定义的元素。当然,在选项板视图 226 中也可以提供或者访问任何其他种类和类别、或子类别的预先定义的元素。在该示例中,致动器可以包括阀门和其他致动器元素,而计算和控制元素可以包括任何与控制相关的元素,例如控制器、诸如 PID 控制回路或其他类型的控制回路之类的控制回路、功能块、控制模块等的指示。如图 15 所示,处理元素可以包括罐、反应器、混合器或者以某种方式来处理物质的其他元素,以及任何其他类型的设备、单位等。属性和测量可以包括框或显示元素,它们被设计为用来显示属性或者测量,或者运行时环境内的其他数据,例如过程变量、告警,等等。形状可以是图元或其他预先定义的形状,而 UI 控制可以包括各种用户界面控制元素,像按钮、滑动块、旋钮、工具框,等等,用户可以在显示屏上操纵这些用户界面控制元素,以实现对该显示的输入。当然,用户定义的元素可以包括任何被预先定义的元素,例如任何其他图形元素或者由图形元素制成的更高级的元素。在一种情况下,用户定义的元素可以包括过程单位、加工区域、或者其他更高级的过程实体。如上所述,可以基于使用编辑器 50 的用户的身分识别,来限制对用户定义的元素访问,或者也可以进行全局访问。当然,应当认识到,任何其他的元素、形状等可以被放置在选项板部分 226 内任何适当的标题下,以便使这些元素能够由用户进行组织,并且易于对其进行访问。

[0121] 当用户将诸如搅拌罐之类的元素放置到主编辑画布 224 中时,可以利用该元素的默认显像来在画布 224 上呈现该元素。在这种情况下,图形分级部分 230 将会显示该元素,并且会采用类似于图 5 中所描述的方式,为该元素提供子元素的分级,例如与该图形元素

相关联的显像、动作、图元,等等。更进一步地,当在包括有可以被绑定到运行时环境上的显像或动画的画布 224 上显示该图形元素时,绑定预订部分 234 将会图示当前所定义的绑定。

[0122] 如果需要的话,通过以任何期望的方式来选择不同的显像,用户可以选择或改变主编辑画布 224 中或者选项板部分 226 中的图形元素的显像。利用下拉列表或下拉框,该下拉列表或下拉框可以通过例如对该元素上的鼠标指针进行右点击来进行访问,或者通过选择分级部分 230 内的不同显像,或者以任何其他期望的方式,用户可以执行该显像选择功能。图 16 示出了屏幕 220 中主编辑画布 224 和选项板部分 226 的一部分,其中将立式罐元素 240 表示为从选项板部分 226 放置到主编辑画布 224 内。立式罐元素 240 是选项板部分 226 中描绘的模板或类立式罐元素 241 的副本或实例化。然而,立式罐 240 的额外显像,包括当鼠标指针在显像 240 上时通过右点击鼠标来访问的在侧边工具条 242 中显示的显像 1-8,能够被选择为当处于在编辑画布 224 中创建的显示之中时要被用作为立式罐元素 240 的显像。如选项板部分 226 所示,通过在选项板部分 226 中的模板立式罐元素 241 上进行右点击,可以获得类似类型的侧边工具条显示 243,从而获得或查看该模板罐元素 241 的可能显像。在侧边工具条 242 中选择不同的显像会改变在画布 224 中使用的罐元素 240 的显像,而在侧边工具条 243 中选择不同的显像会改变存储在选项板部分 226 中的用于模板罐元素 241 的默认设置或显像。

[0123] 除了从选项板部分 226 中选择图形元素和其他元素来创建显示之外,用户还可以从图 15 所描述的工具栏 228 中使用或选择项目,以获得基本绘图工具,例如线,像正方形、矩形、圆、五边形等的形状,文本,等等,并且可以使用这些简单的绘图工具或单元来绘制线条,或者在该显示中添加文本。另外,用户可以利用工具栏 228 中所显示的工具栏连接器元素 245,来将连接器元素添加到画布 224 内的显示中。当被选择时,连接器元素 245 可以向用户提供连接器列表,以便在利用如下拉菜单、对话框等的显示中使用。可能的连接器元素包括管路、传送带、电线、液体流管线或其他类型的连接器,这些连接器实际上将诸如罐、混合器、泵等的一个硬件元素连接至诸如阀门、传感器等的另一个硬件元素上。这些连接器元素,以及在诸如阀门罐、泵等物理设备的不同表示之间提供连接器元素,在美国公开文本 No. 2004/0153804 中进行了更为详细的解释说明,现特意将其合并于此以供参考。该连接器元素可以允许用户与所创建的显示内的不同元素相互连接,并且为如以上对图形元素所解释说明的不同元素定义的连接器点进行配对。例如,管路连接器可以被用来连接具有管路连接点点的不同元素,而管道连接器可以被用来连接已用管道连接器点定义的元素。如果需要的话,编辑器 50 可以实施连接规则,该连接规则仅允许用户通过像由阀门和罐的连接点所定义的适当类型的连接器,来连接不同的图形元素,例如阀门和罐。当然,这些连接元素的外观可能会基于连接类型而有所不同,以便为通过连接元素创建的显示提供更好的观感。

[0124] 当然,工具栏 228 可以包含其他向量绘图工具和标准命令或功能,例如典型的文件 (File) 选项 (新建、打开、保存、另存为、关闭、退出,等等),编辑 (Edit) 选项 (例如撤销操作、重复操作、剪切、复制、粘贴、删除),查找 (Find) 选项,等等。另外,工具栏 228 可以包括提供与图形相关的命令的菜单项目,例如可以被用来添加图形元素固有属性的添加属性 (Add Property) 功能,可以被用来添加图形事件的添加事件 (Add Event) 功能,可以允许用户添加显像的添加显像 (Add Visualization) 功能,可以打开对话框以允许用户输

入关于要创建的触发器的信息的添加可视触发器 (Add Visual Trigger) 功能,等等。更进一步地,菜单可以考虑到格式 (Format) 功能 (例如设置相关文本、文本的线条和填充属性、线条与二维和三维图像) 以及形状 (Shape) 功能,该功能可以包括对象的分组或者取消组,从前到后选择对象的次序,垂直且水平地对准对象,等等。工具栏 228 还可以提供这样的功能,允许用户将动画应用于对象,并且这些动画可以被应用于元素的图元,或者应用于对象的一个或者更多元素,例如通过制作元素宽度、颜色、形状等等的动画,以便提供形状的偏斜、旋转或缩放,从而旋转元素或翻转元素,等等。当然,用户可以获取对与该显示内每个元素相关联的动画和动作的访问,并且可以利用适当的对话框或其他编程工具,来改变这些动画和动作。

[0125] 应当理解,用户或设计者可以使用显示编辑器 50,尤其可以与图 15 的屏幕 220 相互联系,以便快速且容易地创建由标准的二维甚或三维图形元素构成的图形显示,该图形元素可以动态地显示测量、致动器和过程设备。同样,在这些显示中支持并可以提供与控制 and 计算相关的静态元素和用户动态图标。另外,例如通过利用带有略微不同的视图或周围环境信息的相同元素,来创建操作员、维护和仿真显示,用户可以定义多个图层,以便处理该图形显示的不同用户的相互联系要求。

[0126] 图 17 示出了处于控制操作员显示形式的、示例性图形显示的显示屏幕 300,该显示屏幕 300 可以利用图形编辑器 50 来创建。该显示屏幕 300 包括反应器元素 310,该反应器元素 310 与阀门元素 312 相连,并且经由在这种情况下是液体流管线的连接器元素 316,与一个或更多泵元素 314 相连。液体流或者流 (stream) 元素 318 对到达和离开屏幕 300 所描述工厂的该部分的原料流提供引用。应当理解,可以通过选择各种图形元素并将它们相互连接到一起,来创建显示屏幕 300 内的图形,因此该显示屏幕 300 可以包括按照以上关于各个图形元素所描述的方式开发的、高分辨率的图形、动画、动作、可视触发器等等。当然,也可以在图形显示级添加额外的图形,包括动画、动作、可视触发器、液体流元素,等等。通过这种方式,可以将动画和用户动作并入到图形显示中。例如,利用这种能力,可以为过程设备修改静态图形部分,以表示该设备的状态,例如当马达被接通或者跳闸 (trip) 时,阀门的操作状态或位置,等等。另外,动画可以被用来表示与该设备相关联的动态数据,例如表示所填充罐的液面 (例如,使用填充技术),或者通过表示运动的显示变化 (动画) 来表示搅拌器的状态。同样,可以将数据显示元素 319 置于屏幕 300 中,以便显示在显示 300 外部开发的、但是与显示 300 中各部件的操作相关的过程数据或其他数据。

[0127] 如果需要的话,图形显示 300 还可以包括动态图标 (dynamo) 或用户界面按钮 320A 和 320B,这使得用户能够以某种方式与显示 300 交互,在该方式下,允许用户查看附加信息,或者通过与显示屏幕 300 的交互,来对运行时环境采取行动。在某些情况下,可以使用如上相对于图形元素而被描述的可视触发器或动作来执行这些交互。例如,按钮 320A 可以向用户提供关于冷却塔组成的信息的进一步查看,而按钮 320B 可以提供反应器 310 的面板显示。因此,尽管显示 300 包括一组元素,这些元素示出了利用三维部件构成过程或者过程一部分的多台设备,在过去,这可以基于从大量不同源输入的图形,例如包括 In-Tools、Auto-Cad、诸如 Visio、向量绘图、JPEG 和 bitmap 图像格式之类的 Windows 图元文件;现在屏幕 300 可以包括动画,以便在元素级执行例如旋转、度量尺寸、缩放、偏斜、颜色变化,等等,从而提供更有趣和逼真的动画,并因此提供更容易理解的显示。

[0128] 更进一步,可以在显示 300 中示出诸如旋钮、拨号盘、滑动条和按钮的基本用户界面部件,并且可以将这些基本用户界面部件动态地链接到控制系统或其他运行时环境内的信息或控件上。数据视图元素或动态图标还可以提供或图示与控制功能、告警、事件等中的测量相关联的关键参数。例如,动态图标可以被用来显示具有与控制回路参数和工程单位相关的信息的另一个屏幕,并且可以紧接在该动态图标之后进行显示,以便提供该动态图标所显示的值的场景。通过颜色变化,例如控制参数值的背景颜色的变化,可以在该动态图标中反映与控制回路相关联的过程告警的状态。同样,为了消除该显示中的混乱,可以通过颜色变化来表示回路不在所设计的正常操作模式中的事实。当然,也可以使这种动态图标遵循任何标准。

[0129] 在一个示例中,当操作员访问用户交互式部件或动态图标时,脚本或其他程序可以弹起另一个屏幕或显示,例如面板显示或控制面板显示,在图 18 中示出了这些显示的例子。例如,当访问用户交互按钮,例如在屏幕 300 中示出的按钮 320B 时,可以向操作员显示用于反应器 310 的面板,从而操作员可以使用该面板来修改或查看有关反应器 310 的细节。在图 18 的示例中,面板信息 350 与用于反应器 310 的控制回路(被称作 FIG2\_28/TC2-1)相关联,用户可以通过屏幕 300 中的按钮 320A 来访问该反应器 310。利用面板 350 的部分 350A,用户可以利用按钮 352(其操作由动作例程来定义)来改变模式(从级联到例如手动或自动),可以在滑动块显示 354 上查看操作参数的当前值,可以利用箭头 355 来改变与控制回路相关联的设定值,等等。另外,可以在显示部分 350B 中向用户提供有关该控制回路的限制和调谐参数的信息,并且可以在部分 356 中实现仿真能力。如果需要的话,可以通过选择部分 350A 上按钮 358 中的一个,从部分 350A 的视图中获取部分 350B。同样,通过其他按钮 358,用户可以访问有关该控制回路的更进一步的信息,例如趋向数据、诊断数据,等等,或者可以访问和运行控制和诊断程序,例如回路调谐程序。因此,响应于在屏幕 300 上采取或被容许的用户动作,可以通过显示 300 来访问任何其他的活动、屏幕和动作。

[0130] 在控制系统支持在类似的几台设备的定义中使用别名的地方,也可以将动态显示组件设计为支持基于在显示屏幕 300 中所选的那台设备的动态引用。在这种情况下,可以使用预先配置的别名和属性代替通常被定义为显示对象一部分的对象标记或图形属性。这种别名使用能力支持高度的灵活性和可重用性,这是因为类似的显示对象可以连接到不同的 I/O 点,并且代表不同的图形属性、外观和安全性。这种能力可以消除为工厂内几台不同的重复设备重新建立类似显示对象或类似显示的需要。以这种方式,相同的图形显示可以被用来查看工厂内构成和使用上都相同的不同硬件元素。

[0131] 当然,图 17 的显示屏幕 300 可以设计为支持工具栏,包括除了在屏幕 300 的顶部示出的工具栏以外的,或者替代它的水平(在显示以下)和竖直(在显示的右边)工具栏。如果需要的话,可以提供默认工具栏以支持时间和日期显示,通过直接访问要求对告警应答或对告警保持静默的告警显示来查看告警列表,浏览告警概要显示或菜单、主菜单,或者其他标准菜单,或者显示、系统状态显示,等等。

[0132] 图 19 图示了与高级图形显示相关的显示屏幕 400,即具有更多元素、连接、用户界面动作、动态图标和其他数据引用的图形显示。详细地,屏幕 400 图示了一个石灰窑单元的操作,其中空气和其他燃烧燃料产物通过泵 412、阀 404 和相关的连接元素被泵浦或馈送给窑桶干燥炉 408 的输入端 406。同样,从薄板传送带设备 410 馈送罐 409 中的过程原料,

其中薄板传送带设备将石灰运送穿过桶干燥炉 408。当然,在屏幕 400 中所示这些单元和其他设备中的每一个都可以是图形元素,如上所述它们可以单独创建并且放置在屏幕 400 内。桶干燥炉 408 的输入端 406 可以包括动画显示的图形元素,其将干燥炉 408 运转时的火苗或火焰 415 示出为动画,以便清楚地向屏幕 400 的查看者表示窑炉单元的操作。另外,在干燥炉 408 中各个点的温度可以用动态图标或温度显示框 416 来指示,并且如果需要的话,可以通过火苗动画的颜色、干燥炉 408 的颜色,或者以任何其他方式来指示这些不同点的温度。同样,如图 19 所示,由于参数框内的变量值可以与过程控制系统内的特定引用相关联,或者可以从过程控制系统内的特定引用中获取,因此可以使用参数框来图示其他过程参数,例如炉罩压力、窑炉速度、空气总量、甲醇、输入、初级和次级空气输入、原油和天然气输入,等等。当然,在屏幕 400 中图示了还石灰窑单元的其他物理部件。

[0133] 然而,可以在屏幕 400 上提供各种其他信息,这些信息可以从运行时环境内的其他数据源获取,例如处理来自控制系统或者来自控制系统内各设备的数据的其他应用程序。这些应用程序可以包括,例如控制应用程序、维护应用程序、诊断应用程序、商务应用程序等。例如,显示系统温度(度)相对于时间的趋向图 420 显示在显示屏幕 400 的上部中央,并且与数据历史记录或趋势应用程序相关联,并自动地绘制该数据。同样,提供窑炉 412 操作的概要的图表 422 显示在屏幕 400 的上部左侧,图表 422 可以由追踪商务各个方面的商务应用程序提供,例如该窑炉的利润、能量使用、产率等。更进一步地,通过选择位于显示屏幕 400 上部的用户界面按钮 424,可以对可能对用户有用的其他信息进行访问,例如历史趋势、帮助信息、窑炉信息、控制约束、窑炉能量和其他信息。更进一步地,可以在屏幕 400 的底部提供告警旗标或其他旗标,并且可以将其与告警应用程序相关联。当然,利用上述显示编辑器 50 的特征,可以将显示屏幕 400 的所有这些元素和特征配备并编程到该显示当中。另外,可以按照任何期望的方式,将这些和其他特征并入到图形显示中,以创建任何期望类型的显示。更进一步地,可以在屏幕 400 中配备任何期望的动画和图形动作,以便向用户提供更有用或更容易理解的可视信息,从而允许用户更直接地从屏幕 400 查看其他相关信息,等等。

[0134] 因此,如图 19 所示,可以在显示屏幕 400 中显示来自各种不同数据源的信息,这些不同数据源包括由控制子系统访问的过程元素,维护、控制、诊断、调谐和商务应用程序形式的应用程序;来自诸如数据历史记录器的数据库的历史数据或趋势数据,或任何其他数据源。更进一步,通过到这些显示的用户链接,可以从显示屏幕 400 上直接访问其他信息或显示。以这种方式,屏幕 400 所示的图形显示以可能更有用的方式,向用户提供更多信息。

[0135] 由于编辑器 50 可以利用公用的一组图形单素、连接单素等来创建所有的图形显示,因此显示编辑器 50 可以用来容易地创建类似的或相关的显示。这些显示可能涉及并且示出相同的一组工厂硬件,但是却出于不同的目的,例如控制操作员目的、仿真目的、维护目的等,图示该硬件的不同信息。在这种情况下,可以使公用显示或基本显示示出与工厂或工厂一部分相关的硬件元素,而利用该基本显示可以创建不同的显示,为不同用户或不同类型的用户提供不同的信息。

[0136] 作为例子,图 20A-20E 示出了示例性显示,包括为石灰窑单元创建的操作员视图、工程师视图、管理者视图、仿真视图和维护视图,这些视图示出了相同配置中的相同基本



硬件元素,并且使用相同的显像,但是具有为不同功能目的而添加的不同信息。因此,图 20A-20E 中的每一幅显示均具有相同的观感,因为它们是利用相同的基本图形元素由相同的显示编辑器制成的,因此容易在保留对所显示信息的理解与它如何与工厂内的硬件相关联之间进行导航。

[0137] 详细地,图 20A 图示了窑炉单元的操作员视图 500,该窑炉单元具有石灰和泥浆馈送源 502(它可以是流元素),将原材料通过泵 504 馈送给窑干燥炉 508 的冷端点 506。传送带 510 移动来自窑干燥炉 508 的热端点 512 的处理过的石灰,并将这些石灰存放在罐 514 中。产品线或流元素 516 表示排出罐 514 的产品(石灰)数量。同样,可以将来自燃料源流指示器 520 的燃料通过阀门 522 提供给窑干燥炉 508 的热端点 512,所述阀门的颜色可以用来图示阀门的操作。抽风机 524 将空气通过风道连接器元素 523 泵到窑干燥炉 508 的热端点 512,在那里将它与燃料混合。可以在窑干燥炉 508 的热端点显示诸如火苗或火焰 528 的动画,以图示窑干燥炉 508 的操作状态。同样,诱导抽风机 530 通过另外的通风管道系统 532,从窑干燥炉 508 的冷端点 506 抽吸空气,经将该空气送到如烟气流元素 534 所示的烟囱。更进一步地,图示了各种变量或参数框,以表示各种过程参数的值,例如系统各个部分的温度、窑干燥炉 508 的装桶(barrel)速度,空气和燃料流量等等。可以看出,该硬件和这些变量框在图 20A-20E 的屏幕上从头至尾都是共同的,并且提供了显示的基本元素,从而使这些显示具有相同的外表和感觉。

[0138] 然而,图 20A-20E 的各个屏幕均包括适应于工厂内不同功能的附加信息,用户界面按钮和动作。例如,图 20A 的屏幕 500 是操作员视图,并且图示了由框 540 中的实验室测试所测量、估计和提供的残留碳酸盐,框 542 中该窑炉的总能量和比能量,以及由模型预测控制例程提供的显示温度的过去和将来趋势的趋向图 544,这些对于实现最优操作而言非常关键。另外,屏幕 500 提供各种用户界面按钮或动态图标 548,以便允许用户查看关于相关项目的附加信息,例如控制回路信息、元素的更多操作信息,等等。

[0139] 图 20B 的屏幕 550 图示了一工程视图,它非常类似于图 20A,但是它通过虚线 552 示出了对控制很重要的测量位置和信号路径,以便展示基本控制系统和模型预测控制系统如何共同工作来优化操作。图 20C 的屏幕 560 提供管理视图,尽管示出了窑炉单元的基本操作,但是不提供允许获取关于系统各个部件或控制回路的附加信息的用户界面按钮。更进一步地,管理视图 560 包括图表 562,它示出了在过去一天和一个月里,来自实验室测量的总能量、比能量和残留碳酸盐(未转化的进料)。举例来说,可以从在商务计算机上运行的商务应用程序来提供该信息。

[0140] 更进一步地,图 20D 提供一仿真视图 570,它允许仿真器改变仿真系统内的参数,并查看仿真结果。举例来说,视图 570 可以用来训练操作员、测试操作的不同未来模式,等等。如图 20D 所示,仿真器可以通过屏幕 570 上的一个或更多用户界面按钮,改变经由一个或更多对话框 570 提供给仿真操作员的各个参数。过程仿真视图 570 可以用于离线情形以进行训练,或者用于在线情形以提供可能对检测将来的问题有用的追加信息。由于在操作员图形的构造中使用具有仿真能力的智能对象或图形元素,设备类型及其连接是已知的,因此可以由这些图形来生成过程仿真。这些智能对象在美国出版公开 No. 2004/0153804 中进行了更为详细的描述。

[0141] 同样,图 20E 图示了维护视图 580,它提供关于窑炉单元内设备状态的信息。在视

图 580 中,可以利用设备健康指标 582 和 584 来显示设备故障。在这种情况下,指标 582 和 584(在图 20E 中未标注所有这些指标)是半圆,这些半圆带有显示或规定设备当前所监控健康状况的填充颜色。在视图 580 中,指标 584 表示设备具有不及最优的健康状况。当然,可以从维护或诊断应用程序来提供这些指标 582 和 584。更进一步地,应当理解,当检测到过程问题时,维护技术人员可以使用屏幕 580 来深入挖掘(drill down)和寻找解决该问题的推荐程序。作为例子,一个堵塞的滤泥器可能需要进行冲洗,这将要求切断进料,并将该窑炉置于空闲模式,然后可以通过图 20A 的操作员视图 500 来完成这些操作。操作还可以知道对过多砂砾的进料质量进行检查,这可能由上游过程问题而引起。因此,如该简单例子所示,通过在具有相同外表和感觉的不同视图之间进行切换,可以简单且容易地完成不同功能之间的各种交互,例如检测问题并解决它,从而容易进行导航。当然,图 20A-20E 各个屏幕中的信息可以在显示上进行分层,并且如果需要的话,也可以基于屏幕的用户单独地显示。同样,为各个元素或智能对象显示的信息也可以随着用户或用户的身份而发生变化。

[0142] 更进一步地,可以为图 20A-20E 的显示提供附加显示,这些附加显示可以与图 20A-20E 的显示相关联。举例来说,这些显示可以包括,诸如图 21A 和 21B 的控制配置显示。图 21A 的控制显示 585 图示了这样一种方式,通过这种方式将各种控制信号发送给图 20A-20E 所描述的窑炉单元的各个硬件元素,或者从图 20A-20E 所描述窑炉单元的各个硬件元素发送给控制器,以及还图示了与这些控制元素相关联的标签或变量名称。图 21B 的控制显示 590 图示了一控制模块,示出了实现图 20A-20E 的窑炉单元的的控制的控制例程。尽管没有具体地如此显示,但是可以使用屏幕 20A-21E 的相同元素(具有与这些相关联元素的相同或不同显像),或者也可以使用不同的元素,来制作图 21A 的控制显示 585。同样,由于可以使用显示编辑器 50 来制作图 21A 和 21B 的显示 585 和 590,因此这些显示可以配备有上述讨论的任何一种图形和动画能力。更进一步,举例来说,可以通过图 20A 的操作员视图 500,通过在其上配备的一个用户界面按钮 548,来访问控制显示 585 和 590,从而允许操作员从操作员视图 500 容易地获得对当前控制设置的访问。

[0143] 因此,通过以上提供的讨论能够理解,可以由编辑器 50 创建各种相关显示,并且这些显示可以按照许多方式进行分层。特别地,可以如上所述对显示进行分层,以便提供相关或类似的操作员、商务、仿真、维护和工程视图,并且这些视图可以容易地进行相互访问。更进一步地,可以按照能够反映工厂内特定分级结构、逻辑或物理结构的方式,将不同的显示进行分层或者连接到一起。因此,举例来说,可以为工厂内的不同物理区域或地理区域制作显示。因此,可以创建表示工厂内主要区域的单独一个显示,还可以创建例如逐个单元地表示工厂内各个区域内基本结构的附加显示,同时还可以为各个单元创建进一步的显示。以这种方式,用户可以在这些显示中进行深入挖掘,以便获取与工厂中越来越小的部分相关联的越来越多的细节。作为另一个例子,通过用户界面按钮,可以将用于工厂不同区段(section)或不同部分的显示串联或相连在一起,以便操作员可以容易地向前和向后滚动不同的显示,以查看不同的但物理上相连的工厂区段或者逻辑方式的工厂部件。

[0144] 图 22 的示意图更详细地示出了这些类型的显示分层。具体地说,全面或总体的工厂概貌显示 600 可以提供图示整个工厂的基层结构或高层结构的显示,尽管它可能不提供有关工厂内任何特定区段或特定部分的许多细节。通过显示 600,用户能够选择(利用界面按钮)或深入到工厂的任何特定区段或区域,以便拉出工厂内区域 A 的一个或更多的显示

602,或者区域B的一个或更多的显示604。如图22所示,区域A可以具有n个与之相关联的单独显示,所有这些单独显示都在逻辑上彼此串联在一起,以便反映通过该工厂区域的过程流,或者与工厂区域A相关联的某些其他逻辑结构。标记为602a、602b、……、602n的n个区域A的显示,可以利用向前翻页或向后翻页型的动作进行访问,其中用户可以从一个显示滚动到下一个显示。以这种方式,用户可以按照对操作员有意义的方式,容易地从区域A的一个部分滚动显示到区域A的另一个部分。更进一步,如图21所示,操作员可以从单独的各个显示602a-602n中获取预定义的信息或其他显示信息。因此,当查看区域A的显示602a时,操作员能够获取另一个显示610,该显示描述了显示602a中关键参数的预定义趋向。同样,当查看显示602b时,操作员能够访问具有启动和关闭程序信息的列表或文档612。当然,用户可以在显示602a、602b等之间,向前和向后滚动。

[0145] 同样,当查看图示为包括m个显示604a-604m的区域B时,用户可以使用显示内的向前和向后(前进和后退)按钮,在相同细节级的显示之间滚动,从而使操作员或其他用户有可能访问包含有过程所显示区段的上游或下游信息的显示。另外,可以配备动态图标或其他用户界面按钮,以允许对另一个显示进行访问,从而获取与当前显示内的元素相关联的其他信息。通过使用这些工具,有可能创建一显示分级结构,通过该显示分级结构,概貌显示可以用来访问各个过程区域中的关键显示。

[0146] 另外,如图20A-20E所示,可以对显示进行分层,以便对于工厂的相同区段或相同部分存在多个显示,但是它们可以用于不同的功能目的,例如用于操作员动作、维护动作、仿真动作、商务动作、工程动作,等等。在图22中将这些单独的功能显示示为位于显示602a、602b等之下的层,并且如果需要的话,可以相互进行访问。因此,当操作员或用户查看显示602a时,用户可以在用于工厂该区段的其他功能显示之间进行切换,或者访问用于工厂该区段的其他功能显示(如果准许这些访问的话),例如维护视图、仿真视图、商务视图,等等。当然,以使用相同的基本显示元素的方式对维护视图、操作员视图、仿真视图等进行分层,能够提供相对于工厂内不同功能的在这些视图之间更容易的切换,以及对工厂内所发生一切的更好理解。

[0147] 为了实现该功能和其他功能,可以将图形显示创建为包括该图形显示预期角色(或功能)的指示。这些角色(role)可以包括,例如面板显示、细节显示、主控制显示、示意显示、维护显示、商务显示、仿真显示,或任何其他用户定义的角色。该角色是图形显示的一部分,并且当把图形显示分配给控制模块或硬件设备以便定义在运行时间对该显示的使用和访问时,可以使用该角色。分配给图形显示的角色指示可以用来指示将特定显示分配给配置系统或工厂中的哪些地方或哪些区段。另外,该角色信息可以用来基于特定用户所执行的工作,确定该特定用户可以访问哪些显示。例如,控制操作员不能够查看或访问定义为仿真或商务显示的图形显示。

[0148] 上述图形显示能力还可以用来创建专用显示,以允许容易地对极关键设备的状态进行监控。这些类型的显示的一些例子包括关于过程关闭、振动监控、燃烧器管理、吹灰器操作、和安全系统状态的首先输出指示(first out indication)。当然,可以构造相关的显示,以便概括该信息,并且在例如吹灰器的运动设备包含动画的情况下,动画可以有效地用来允许操作员快速地访问或理解系统的操作。另外,大多数控制系统的计算能力可以用来实现操作成本、效率等的在线计算,并且可以很容易地将这种类型的信息并入到操作员图

形显示中,从而操作员可以使用该信息来改进过程操作。同样,各种技术可以用来集成控制系统中的子系统信息,以便标准图形和动态图标可以用来创建操作员显示,以便允许从更高级显示对子系统信息进行访问。在某些情况下,可以执行矩阵值的 3D 绘图,以便显示信息(例如片规信息)。

[0149] 另外,应当注意,可以利用例如 Microsoft Avalon 控件提供的那些向量图形,有利地实现图形元素和图形显示,以便提供灵活性和速度。向量图形的使用可以提供使用可缩放图形的能力,从而能够对所创建的显示进行成比例缩放和改变尺寸,以便适应特定的显示机器,以及提供使用缩放比例的动画。

[0150] 一般而言,正如在标题为“各个过程图形显示及其包含的图形显示元素的创建(the creation of each process graphic display, and graphic display element contained therein)”的共同未决申请中详细描述,每个过程图形显示及其包含的图形显示元素的创建,都记录在以第一说明性格式阐述的各自文本的说明中。不管要再现的图形的复杂程度如何,各个说明中的脚本命令均提供高效的、无存储器的集约(intensive)机制来定义显示。因此,该说明性格式和脚本命令,可以基于大量不同标记语言中的任何一个。更具体而言,可以依靠基于 XML 的标记语言来阐述各个显示和显示元素的再现定义(也称作 XML blob),并且该 XML blob 可以用来在将其下载到过程运行时环境之前,将这些显示和元素存储在配置库或数据库中。为了支持高级图形,例如动画,标记语言还可以根据向量图形方案来定义这些图形。

[0151] 如上所述,正如在标题为“加工厂用户界面中基于标记语言的动态过程图形(Markup Language-Based, Dynamic Process Graphics in a Process Plant User Interface)”的共同未决申请中所描述的,该申请合并于此以资参考,将过程图形的动态特性设计为当在线条件或仿真条件发生变化时,反映加工厂参数的当前值。为此,可以将过程图形链接至反映该变化的数据源。每个基于 XML 的描述都可以因此包括一个或更多数据源参考,这些数据源参考通常识别要根据该数据进行修改的各个动态图形参数(例如罐内的颜色变化)的数据源位置。在通过编辑器进行配置期间,还可以开放该数据源位置以用于稍后的规定,从而该脚本识别一个别名或位置标志符,以便引用稍后要规定的数据源或路径信息。由于通过基于 XML 的描述来规定过程图形显示的数据源信息和其他性质(例如,诸如事件处理的行为),所以基于 XML 的语言也可以称作 PGXML,或者过程图形 XML。

[0152] 一旦完成了定义过程图形显示及其组成元素的配置和设计工作,配置工程师或其他用户就可以选择处理 PGXML 描述,以准备将这些过程图形下载到操作员工作站或其他用户显示设备。一般而言,对图形显示和显示元素的每个 PGXML 描述进行处理以生成(i)与要使用的图形再现引擎兼容的向量图形格式,和(ii)带有指令的代码,其用于规定显示的数据源参考和任何其他非图形功能(例如行为)。向量图形格式,脚本也可以是说明性语言、或基于 XML 的语言。在利用 Microsoft Avalon 用户界面架构的实施例中,可以用 Microsoft XML 来阐述向量图形脚本。其他实施例可以使用开放源格式,SVG(可缩放向量图形)。也可以用 C# 或任何其他适当的编程语言来阐述该代码。

[0153] 在某些实施例中,接下来将向量图形脚本与相关代码合并,并编译成阐述用于操作员工作站或其他用户显示设备上的可执行命令的文件。为此,可以为每个过程图形显示和图形显示元素创建各自的动态链接库(DLL)。无论如何,在下载之前可以执行向量图形脚

本和相关代码的编译,从而使网络数据传输要求降到最低。

[0154] 一旦创建图形元素和图形显示,就可以将图形元素和图形显示存储在配置数据库中作为类属对象或模板对象,并且可以在用于运行时环境之前,将其保持为基于类的或基于非类的(non-class based)对象或元素。一般而言,此处讨论的图形元素和图形显示可以是显示模块形式的模块,如美国公开文本 No. 2004/0199925 中所描述的显示模块,其合并于此以供参考。类模块(对象)是不绑定或捆绑到加工厂或过程控制系统内的任何特定硬件或设备上的模块(对象),而是这样的对象,即通过该类对象可以实例化绑定到加工厂或过程控制系统的其他对象。一般而言,为了配置加工厂,特别是配置加工厂的运行时环境,配置引擎,比如图 1 的配置工作站 20 中所示的一个配置应用程序 33,可以用来将图形对象(包括图形元素和图形显示)与加工厂内的其他逻辑实体和物理实体关联起来,包括像控制模块、设备模块(例如单元模块)、过程模块等的逻辑实体,或者诸如操作员工作站或其他显示设备的物理实体。在某些情况下,图形对象可以是类对象,并且可以与像设备模块类对象或控制模块类对象之类的其他类对象相关联,或者图形对象也可以是单独的对象,并且可以与实例化的对象相关联,例如已经绑定到加工厂内设备上的对象,以及已经下载到例如加工厂内的控制器、工作站或其他设备的对象。

[0155] 因此,如美国公开文本 No. 2004/0199925 所述,将其其他类对象存储在库内并在其中进行配置,然后将其下载到过程控制系统的各个部分或子部分中,通过类似于上述方式的方式,可以在加工厂的配置系统内配置以上标识的图形元素和图形显示,并将其下载到加工厂运行时环境内的硬件。详细地,对诸如图 1 的数据库 28 之类的配置数据库内的数据进行操作的配置引擎,使用配置数据库分级结构,如图 23 和图 24 所示的分级结构,使用户能够查看和改变加工厂内的配置。图 23 的配置分级结构屏幕 700 示出了用于控制系统 702(贴有 DeltaV 控制系统的标签)的配置数据库包括库部分 704、系统配置部分 706 和搜索结果部分 708。

[0156] 库部分 704 包括还没有分配给或下载到运行时环境中,相反而是存储为模板对象或类属对象和未绑定对象的类对象和其他对象定义。如图 23 所示,库部分 704 包括控制模块部分 710、设备部分 712、过程模块部分 714 和操作界面部分 716,以及分批部分和安全器具系统部分(未用标号指出)。

[0157] 尽管没有展开控制模块部分 710,但配置数据库的这部分一般包括不同的控制模块,例如控制模块模板和控制模块类对象,它们已经为加工厂创建以用于该加工厂内的各个控制器和其他控制设备。更进一步,设备部分 712 包括设备对象的指示,所述指示可能包括该加工厂内使用的实际设备和各种类型设备,例如阀门、传感器、控制器、工作站等的指示,以及设备类对象、装置类对象等的指示,这些指示定义了与加工厂内不同设备或硬件相关联的逻辑元素。更进一步,过程模块部分 714 包括为加工厂创建的各种过程模块,包括为这些过程模块定义的过程模块模板和过程块定义。这种过程模块在美国公开文本 No. 2004/0199925 中进行了更详细的描述,因此在这里不再进一步描述。

[0158] 然而,如图 23 所示,配置分级结构 700 的操作员界面部分 716 存储与上述图形对象相关的信息。详细地,图形部分 720 包括各种复合过程图形定义、图形类对象和图形模板,它们定义了为该系统创建的各种图形元素和图形显示。应当注意,该复合过程图形定义部分通常包括为系统创建的图形元素,过程图形类部分包括按照以上对类对象所述的

方式创建的图形显示,而过程图形模板可以包括基于非类的图形显示和其他模板对象。存储在配置数据库的该部分中的图形对象一般是未绑定对象,其可以用来创建其他对象,并且可以在配置过程期间使用,以便定义将图形元素和图形显示绑定到加工厂内各个硬件和软件上的方式。更进一步,Operation Interface(操作员界面)部分 716 下的 Layout(布局)部分 722 定义了用于操作员界面的各种布局,而 Roles(角色)部分 724 定义了可以在操作员界面上实现的各种角色,以及由各个图形显示和图形元素履行的角色。

[0159] 因此,如配置分级结构 700 所示的配置数据库可以包括这样一个部分,该部分将包含图形元素和图形显示的图形对象存储为类属模板,存储为类对象或存储为特定或单独的元素或显示,虽然它们并不绑定到特定的过程实体上,但是它们可以用来定义将被绑定到特定的过程实体上的元素或显示。一般而言,配置工程师或其他用户可以使用分级结构 700,按照以下将详细讨论的方式,将加工厂配置为包括图形显示能力。

[0160] 图 24 图示了分级结构屏幕 730,其中已经将系统配置部分 706 展开,以便图示作为系统配置的一部分,过程图形元素和图形显示如何可以与加工厂内的各种逻辑实体和物理实体相关联。特别地,系统配置部分 706 下的过程图形部分 734 定义了各种图形显示,它们通常被定义成下载到加工厂中以用于运行时环境。一般而言,由于过程图形部分 734 应用于整个系统配置,因此可以包括例如面板显示、操作员显示、细节显示、维护显示、仿真显示等的过程图形部分 734 内的图形显示和图形元素,能够下载到加工厂内的各个显示单元(例如,各个工作站或其他显示设备)。然而,图形元素和图形显示可以改为与系统配置的特定部分相关联,所述特定部分包括控制策略部分 736 和物理网络部分 740,它们可以限制在其上能够使用这些图形元素和图形显示的显示设备。

[0161] 一般而言,控制策略部分 736 定义分配给加工厂内不同物理和逻辑部分的各种控制例程,或者定义对加工厂内不同物理和逻辑部分所执行的策略。不同的图形显示(仍然可以是任何类型的显示,例如面板显示、控制显示、维护显示等)可以分配给控制策略部分 736 的特定分部(subdivision)或子类,包括例如分配给各区域(例如分配给名称为 Area\_A 742 和 Area\_B 744 的区域),分配给与各区域相关联的控制模块,例如分配给控制模块部分 746,以及分配给各区域的过程模块,例如分配给过程模块 748。因此,如图 24 所示,显示部分 750 与 Area\_A(区域 A)部分 742 相关联,它定义了用于工厂 Area\_A(区域 A)的以及工厂 Area\_A(区域 A)中的所有显示。尽管未展开,但是文件夹 750 下的显示可以包括与 Area\_A(区域 A)中不同组的硬件相关联的各种显示,例如图 22 的不同显示 602a-602n,以及可以从这些显示进行访问的各种子显示,例如图 22 的显示 610 和 612。为这些显示定义的角色可以表示这些显示能够彼此进行访问,或者能够由使用显示设备的操作员进行访问的方式。因此,如对于图 22 的描述,用户可以在部分 750 中的各个显示中进行滚动,以便查看工厂的 Area\_A(区域 A)的不同部分,或者查看关于任何特定区域的更多细节,包括作为更高级显示中的许多细节的、未示出在更高级显示中的特定部分的更详细显示。

[0162] 更进一步地,如控制模块部分 746 下的显示图标 752 所示,可以将显示分配给 Area\_A(区域 A)部分 742 内的特定控制模块。在这种情况下,显示 752 与控制模块回路(名称为 LOOP)相关联,具体地可以是与该特定控制回路相关联的控制操作员显示。为显示定义的角色也可以表示它在作为控制显示一部分的操作员站内的角色。更进一步地,如关于 Area\_B(区域 B)部分 744 所示,显示 754 通常可以与该区域相关联,而显示 756 可以与分

配给某一区域的特定过程模块相关联。在这种情况下,显示图标 756 与名称为 PMOD2 的过程模块相关联,其中 PMOD2 是与 Area\_B (区域 B) 中的设备相关联的过程模块。例如,PMOD2 可以是单元模块或设备模块或任何其他类型的过程模块。

[0163] 因此,如通常在控制策略部分 736 之下所示,由于在配置系统内创建了这些控制定义,所以显示可以与特定控制策略或控制定义相关联。如图 24 的例子所示,显示可以与基于逐个区域来定义的控制定义相关联,例如用于区域部分 742 和 744 的定义,以及与基于逐个控制模块来定义的控制定义相关联,例如用于控制模块 750 的定义,以及如显示部分 756 所示,与基于逐个过程模块来定义的控制定义相关联。另外,如果需要的话,设备项目可以与角色相关联,这允许能够动态地从设备元素对与这些角色相关联的显示进行访问。这些设备角色和显示可以存储在过程图形部分 734 中。

[0164] 另外,显示还可以分配给加工厂内的实际硬件单元,或者与加工厂内的实际硬件单元相关联,所述实际硬件单元包括操作员工作站或其他显示设备。因此,如在物理网络部分 740 以下所示,可以将特定显示分配给特定的操作员工作站,或具有显示器的其他运行时机器。在图 24 的特定例子中,物理网络部分 740 包括控制网络 760,它具有相关硬件,例如名称为 CTLR1 的控制器、可以是配置站的 ProPlus 站,以及名称为 OperatorStation 的操作员工作站 762。定义工厂内特定工作站或用户界面,或者与工厂内特定工作站或用户相关联的 OperatorStation (操作员站) 762,包括告警和事件功能、操作员活动、连续历史记录功能和与之相关联的仿真活动,尽管诸如维护活动、商务活动等或其他活动或功能也可以与任何特定的用户界面相关联,并且能够在任何特定的用户界面上执行。以这种方式,特定的用户界面可以分配给加工厂内的角色,并且具有用于存储在该显示设备中或下载到该显示设备中的那些角色的显示。

[0165] 如图 24 所示,显示 764 和 766 被分配给操作员功能或活动,并被分配给由工作站 762 所执行的仿真功能或活动。尽管在图 24 中未示出,但是也可以将显示分配给其他显示设备,例如与加工工厂内其他节点相关联的那些显示设备,以便在这些显示设备上执行。更进一步,对操作员工作站或其他运行时显示机器内的特定功能的显示分配,可以确定这些显示可以在运行时环境内执行的角色或功能,例如操作员功能、仿真功能、维护功能等。当然,按照由配置系统设置的方式,这些不同的功能可以在相同的显示设备或者不同的显示设备上执行。同样,尽管配置分级结构 730 图示了将显示分配给操作员工作站,但是也可以将显示分配给其他类型的显示设备,包括诸如 PDA 的手持式计算机显示设备,电话设备,商务工作站,或任何其他期望类型的显示设备。

[0166] 配置引擎能够使用户,例如配置工程师,使用配置分级结构,来定义和管理将过程图形显示绑定到加工厂或下载到加工厂内的方式。一般而言,诸如配置工程师的用户,可以按照用户能够对其他对象所操作的相同方式,创建、删除、重命名、分配和下载配置系统内的显示对象。更具体而言,配置引擎能够通过定义将这些显示分配到哪里(给哪些设备)来执行,并且定义在运行期间这些显示应当绑定到的过程实体(逻辑的和物理的),将图形显示集成到加工厂的配置中。如果需要的话,诸如图 23 和 24 的那些配置分级结构屏幕,可以用来指示需要的下载,并用来在这些显示上执行版本控制。

[0167] 配置可能出现在配置系统的多个级。在第一个地方,配置工程师可以首先将库部分 704 中的显示类对象与模块类对象关联起来,将特定的显示类对象绑定到特定的模块类

对象上,例如设备模块类对象,区域模块类对象,等等。其后,当对模块类对象进行实例化并将其分配给加工厂内的特定部分(或进行其他配置)时,将与所实例化的模块对象相关联的显示对象(例如图形显示),连同该模块对象一起绑定到相同的硬件上,由此不需要花费额外的精力就将图形显示分配给工厂内适当的硬件,或者将该图形显示对象下载到适当的硬件上。相反,在这种情况下,图形显示对象遵循它要绑定到其上的模块对象的配置。更进一步,如上所述,当显示类对象与模块类对象相关联时,模块类对象的变化可以自动地使这些变化传播到相关显示类对象内的那些相同元素上,这些变化进而又可以向下传播到图形显示的实际实例化版本上。在一个例子中,如果图形显示包括含有一个称作 Loop 的控制模块的混合器(设备类对象),并且可以选择并重命名该控制模块,则配置系统可以自动地将对图形显示内 Loop 的引用变为新的名称,由此确保使用该新名称来更新图形显示绑定。然而,新近改变的实际显示可能仍然需要下载到运行时机器上,以便在运行时系统中发生变化。如果需要的话,例如通过紧挨着适当的图形显示图标放置一蓝色三角形,可以紧挨着存储在配置分级结构内的显示来放置需要下载的指示。

[0168] 另一方面,可以将图形显示单独或直接分配给加工厂或配置系统内的不同部件。特别地,如图 24 所示,可以将各个图形显示放置在系统配置部分 734 之下的过程图形部分 734 中。例如通过在库部分 702 中进行选择,并将其拖放到部分 734 上,可以将这些显示放置在该部分中。当然,同样也可以使用将图形显示移动到分级结构 730 的特定部分的其他方式。无论如何,部分 734 中的图形显示可以包括,例如面板显示,或者其他基于类或基于非类的显示,并且通常适于下载到工厂内的所有工作站,或者下载到系统配置部分 706 所覆盖的工厂部分。

[0169] 然而,用户可以将显示移动到过程图形部分 734 以外,并移动到特定的子部分(subsection)中,以便控制能够访问这些显示的过程控制系统内工作站(或其他显示设备)的身份识别,从而控制可以在其上使用这些显示的设备的身份识别。为了将特定的显示分配给逻辑或物理工厂部分,用户可以例如选择一图形显示,将该图形显示拖放到配置分级结构 730 的特定部分上,由此定义该图形显示与该显示要与之相关联的逻辑或物理实体之间的关联。基于把显示拖放或关联到其上的配置的那部分,当该配置步骤出现时,可以自动地填充用于显示的解析表。

[0170] 例如,可以将图形显示从库部分 702 拖拽到过程图形部分 706,以便关联该显示与运行时环境。同样,例如过程图形部分 734 中的图形显示,可以移动到其他部分中,例如移动到 Area\_A(区域 A)显示部分 750(使得该显示专用于 Area\_A(区域 A)实体,并且一般只能够在任何 Area\_A(区域 A)设备上实现这些功能),Area\_B(区域 B)显示部分 754,等等。同样,图形显示可以与逻辑实体的子部分相关联,例如与名称为 LOOP 的控制模块相关联(用显示文件夹 752 表示),这使得这些显示专用于该逻辑实体,并且只能在执行该控制回路功能的机器上,或者在定义与该控制回路功能相关联的机器上利用这些显示。另外,可以将显示分配给特定的显示设备或显示设备的功能子部分,例如由 OperatorStation(操作员站)762 内的操作员功能,或者由 OperatorStation(操作员站)762 内的仿真功能 770 定义的那些功能子部分。尽管没有详细示出,但是用户也可以将图形显示分配给区域、小区、单元、设备模块,以及分配给不同的控制部分,以便定义这些图形显示与加工厂相关联的方式,以及这些显示用在加工厂内的方式。



[0171] 当将图形显示分配给逻辑实体,例如分配给控制策略时,与用区域标志如 Area\_A (区域 A) 和 Area\_B (区域 B) 所定义的类似,将图形显示分配给这些逻辑实体所分配的物理界面(例如工作站)。类似地,当将显示分配给单元、小区或设备模块时,将该显示分配给该单元、小区或设备所分配的工作站或其他界面设备。

[0172] 当模块包括多个图形显示时,如当创建这些显示时分配给这些图形显示的角色指示所定义的,该模块可以定义用于这些显示的角色,或者定义每个显示履行的角色。接下来,可以将对任何显示设备内这些图形显示的访问与为该显示定义的角色相互关联,或者由该角色来限定。更进一步地,如果将图形显示分配给某一区域或某一模块,对该图形显示的引用可以作为图形显示,显露在它所属区域或模块的显示文件夹中。一旦将图形显示分配给某一区域或某一模块,该区域或模块所拥有的该显示引用以及该显示就在该区域或模块的任何分配或移动期间,跟随该区域或模块。另一方面,当将区域或模块分配给工作站时,如该显示的角色所定义的,属于该区域或模块的显示将自动分配给该工作站的操作员子系统或功能。当然,通过直接放置到配置分级结构的物理网络部分 740 内的特定界面设备或界面设备的子系统之内或之下,可以将各个图形显示直接分配给显示设备的操作员子系统。

[0173] 因此,可以理解,通过将其拖放到配置屏幕 730 内这些实体的每一个中,可以将图形显示分配给各个节点、控制区域、控制模块、过程模块或其他逻辑控制实体,以及分配给各个显示设备及其子系统。当然,如果需要的话,可以将相同的图形显示分配给多个站,并且可以用分级结构 730 内各个文件夹中为每个显示名称配备的图标来表示每个显示的状态(下载的或刚刚分配的)。如果需要的话,也可以在分级结构 730 使用其他图标,以便指示何时将图形显示锁存在存储器中,以及它们是否可以保存在磁盘上的虚拟存储器中。

[0174] 一般而言,图形显示是由显示内容(例如,定义显示能够可视地出现在显示屏幕上的方式的 XAML 脚本)、本地表和引用构成,其中引用可以是对其它显示的引用,对诸如控制参数等的运行时参数的引用,对显示控制、运行时别名(有些可以在运行时绑定)的引用,本地表引用和全局表引用。一般而言,配置引擎独立于所述引用来处理图形元素(XAML)的图形部分,使其更容易处理和使用图形显示。特别地,配置系统可以在配置过程期间通过根据在配置分级结构中所执行的分配来填充不使用动态别名的引用而绑定图形显示。由此,举例来说,当用户把图形显示从配置分级结构 730 的过程图形部分 734 或从库部分 704 拖动到特定区域或控制模块中时,在所述图形中自动地执行绑定,以把图形显示中的引用绑定到与那个区域或模块相关的特定元素上。以这种方式,基于正在由用户执行的配置,可以自动地执行下载绑定,这减少了用户必须手工执行以规定这些绑定的大量工作。

[0175] 更进一步,如果在配置步骤时有未解析(unresolved)的绑定,那么配置引擎可以询问用户来提供用以解析该绑定的信息,诸如改变该绑定或引用。更进一步地,如果需要的话,批量(bulk)编辑工具,诸如 Microsoft Excel® 的电子表格程序,可以用来帮助配置工程师在配置过程期间填充或定义解析或绑定参数。无论如何,这些配置步骤填充了参照图 13 所讨论的解析表 208,这使得图形显示可以在运行时环境中使用。由此,用户可以按照与在美国公开文本 No. 2004/0199925 中描述的用户对过程模块中所做的方式相同的方式,执行对图形显示(和图形元素)的绑定和下载。

[0176] 当然,用户或配置工程师可以通过点击图形显示并将其从分级结构 730 中移除,

或者结合对话框以使得这个图形显示进行去分配,来去分配(unassign)图形显示。更进一步地,用户可以查看所分配的特定图形显示或一组图形显示的分配方式,而这种信息例如可以通过图 25 所示的对话框 800 来提供。图 25 的对话框指示了两个显示 Display1(显示 1)与 Display2(显示 2)中的每一个在加工厂中进行配置的方式。特别地,对话框 800 指示了使用每个显示的地方(例如,Display1 用在称作 Area\_A(区域 A)的工厂区域和称作 Mod1 的控制模块中)、已经分配了的物理站(例如,Display1 分配给称作 Oper1 的操作员工作站)和显示履行的角色(例如,Display1 是每个使用中的主显示)。

[0177] 可以理解,显示角色可以用任何期望的方式改变,并且该角色可以用来指示配置系统的可以分配图形显示或可以使用图形显示的部分,即用于什么功能。如果需要的话,图形显示可以用于履行多个角色。当然,如上面解释的,当用户对工作站或其它显示设备分配区域、或某种类型的模型时,与这个区域或模块相关联的任何显示也分配给所述工作站,以确保用于那个区域或其它模块的图形支持可以提供在工作站上。而且,当图形显示分配给工作站或其它显示设备时,基于该图形显示的角色,诸如图形显示是否是仿真显示、操作员显示等,可以将图形显示分配给工作站的子系统。另外,用户可以把图形显示拖放到工作站的特定了系统或其它显示设备上,并且如果错误地分配了角色,那么配置系统可以显示对话框,以便向用户指示正在进行错误分配、或者允许用户改变图形显示的角色。

[0178] 在利用图 23 与图 24 的配置引擎与配置分级结构 730 执行适当的或期望的分配之后,用户操作员可以执行图形显示的实际下载,这使得为了在配置系统中定义的目的和角色,将图形显示实际下载到如配置分级结构 730 中所定义的适当操作员工作站。在此下载期间,基于对区域、回路、控制模块、过程模块等,以及操作员工作站或其他显示设备的分配,可以解析显示内的绑定。

[0179] 如果需要的话,配置分级结构 730(图 23 所示)的搜索结果部分 708 可以使用户能够执行对用在整个加工厂或配置系统中的任何单独图形显示或图形显示中的图形元素的搜索。特别地,配置引擎可以包括搜索引擎,当由用户启动时,可以执行对特定图形显示或图形元素的搜索,并且可以把该搜索结果存储在搜索结果部分 708 中,以便向用户提供使用任何特定的图形元素或图形显示的完整列表。这种搜索对进行改变的用户是有用的,能够使这些用户查看哪个图形显示需要进行改变,或者为了实现这些改变需要进行哪些下载。

[0180] 当上述情形实现时,此处描述的任何软件都可以存储在诸如磁盘、光盘或其他存储介质之类的任何计算机可读存储器中,或者存储在计算机或处理器的 RAM 或 ROM 中,等等。同样,该软件可以经由任何已知或期望的传送方法传送给用户、加工厂或操作员工作站,这些传送方法例如包括在计算机可读磁盘上、或者在其他可移动的计算机存储机制上、或者通过诸如电话线路、因特网、万维网(WWW)、任何其他局域网或广域网等的通信信道(这些传送都被视为与经由可移动的存储介质来提供这种软件是相同的或可互换的)。而且,这种软件可以在没有进行调制和加密的情况下直接地传送,或者可以在经由通信信道发送之前,利用任何适当的调制载波和/或加密技术进行调制和/或加密。

[0181] 因此,尽管已经根据特定的例子对本发明进行了描述,但是,这些例子仅仅是示例性的,而不是限制性的,对于本领域的普通技术人员来说,显然可以在不脱离本发明的精神和范围的前提下,对所披露的实施例进行更改,添加或删除。

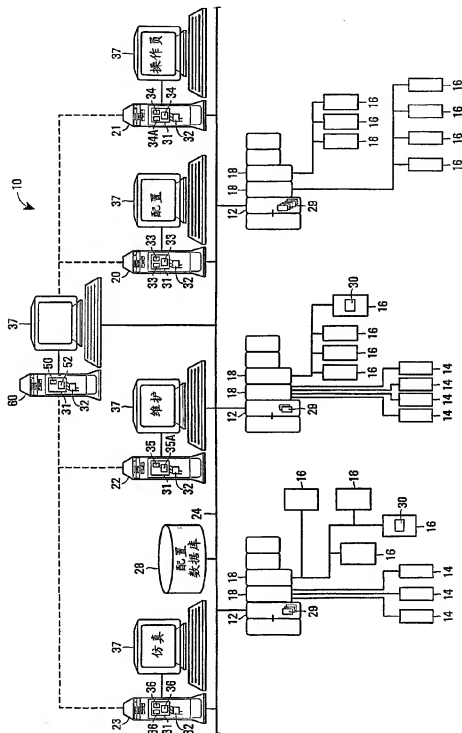


图 1

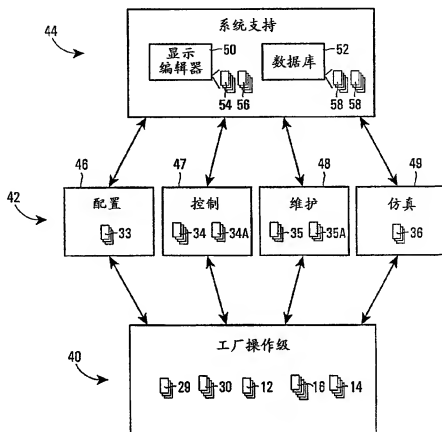


图 2

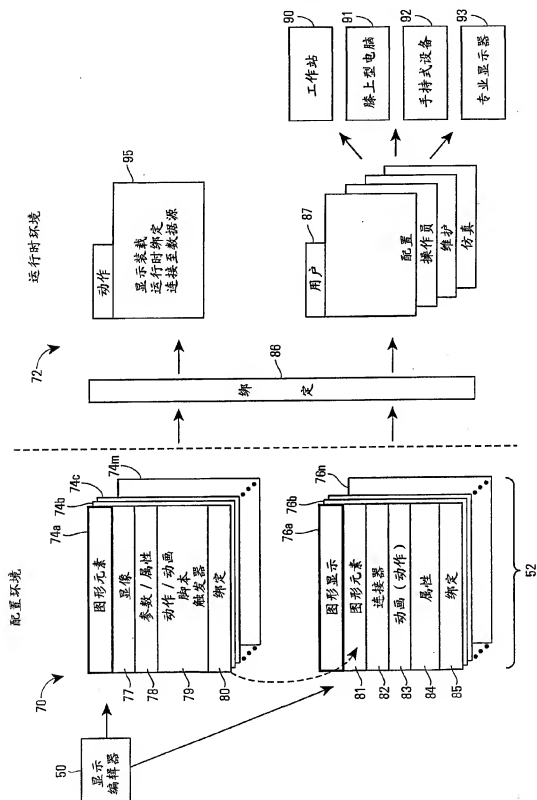


图 3

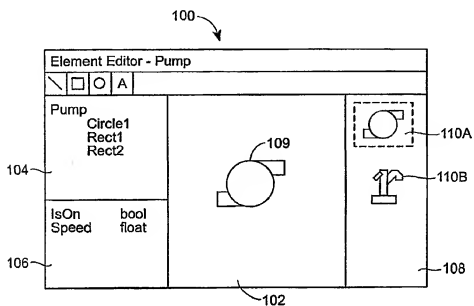


图 4



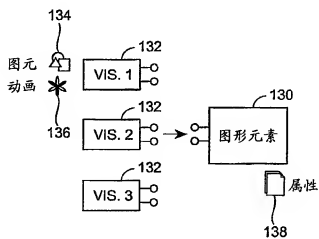


图 6

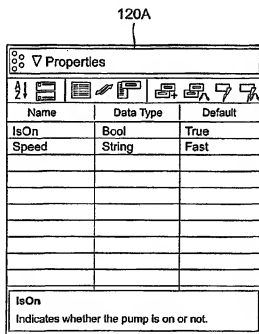


图 7





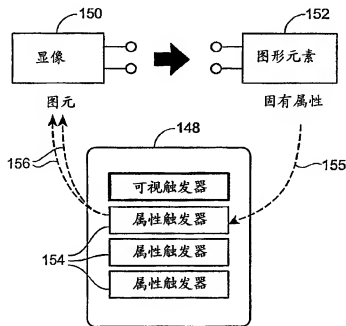


图 10

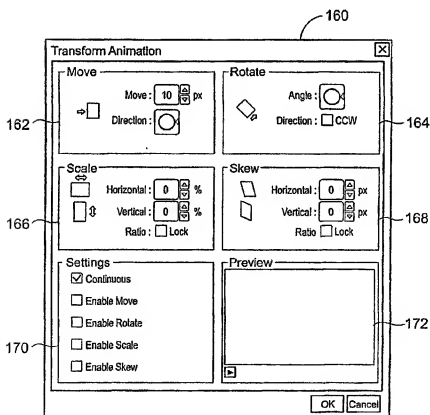


图 11

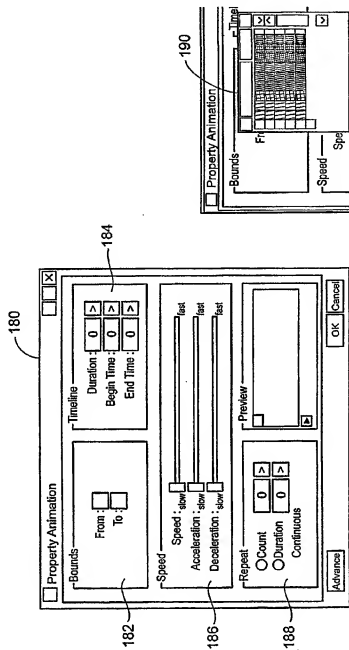


图 12

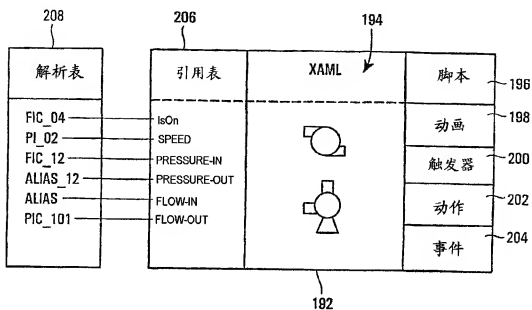


图 13

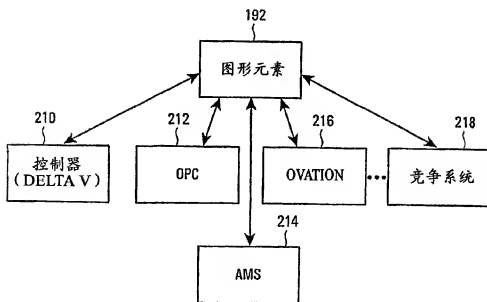


图 14

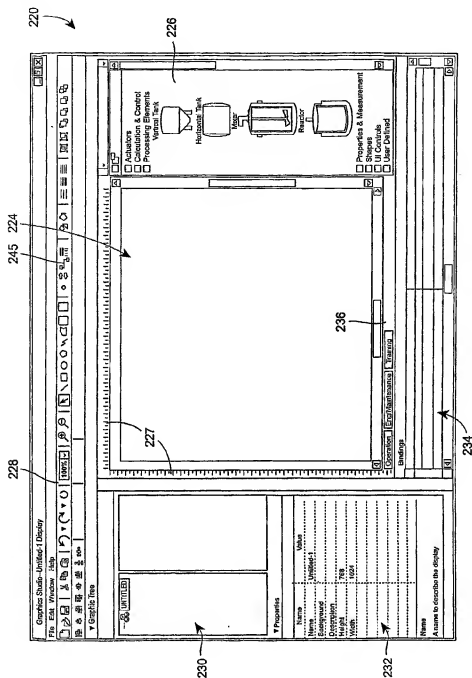


图 15

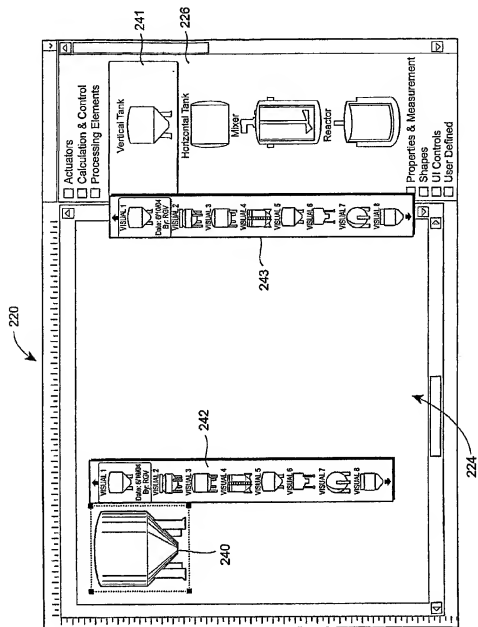


图 16









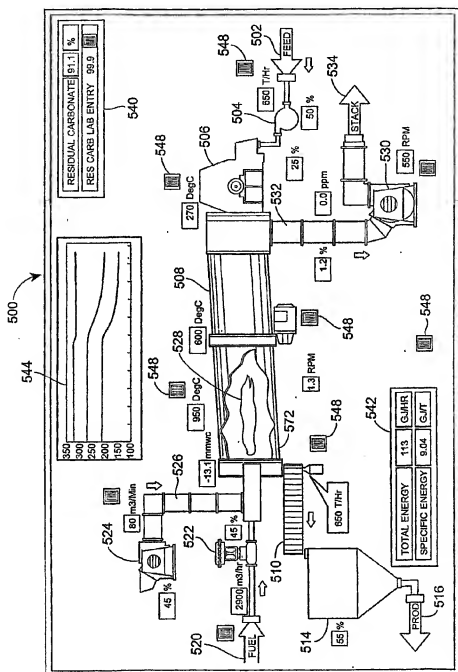


图 20A

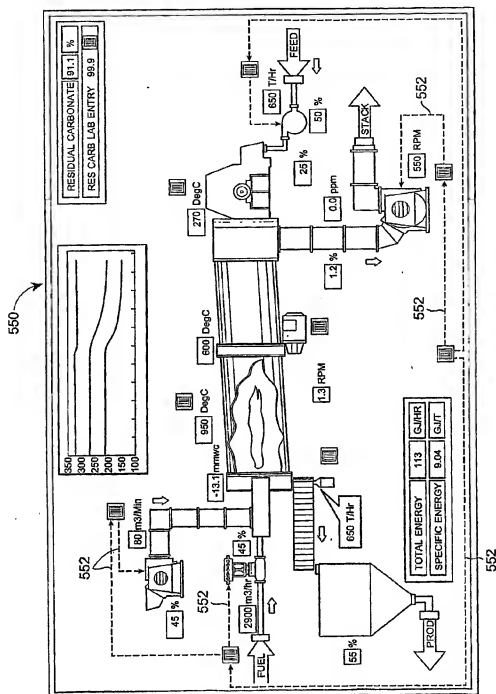
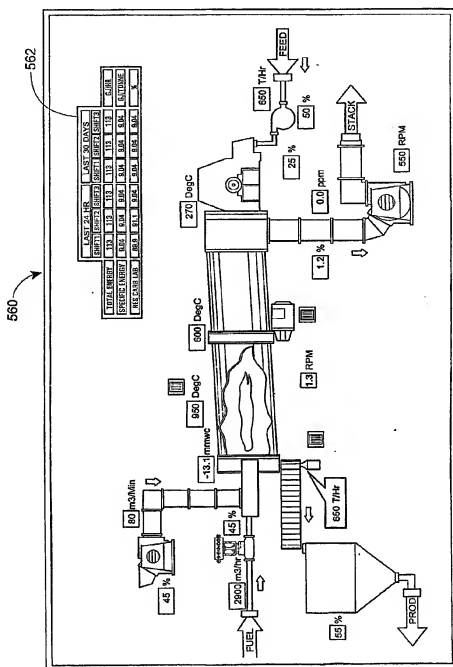


图 20B



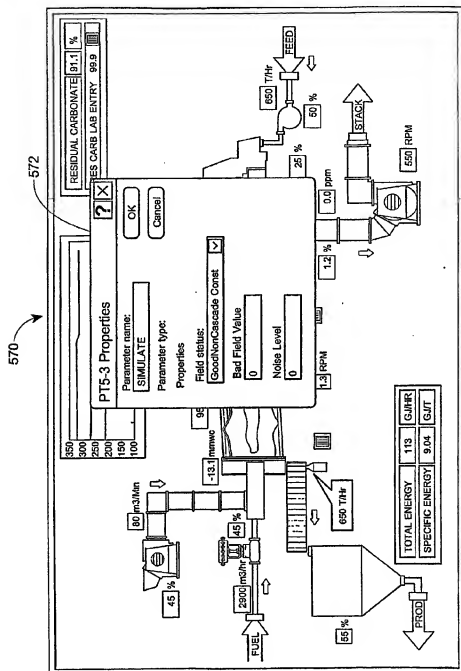


图 200

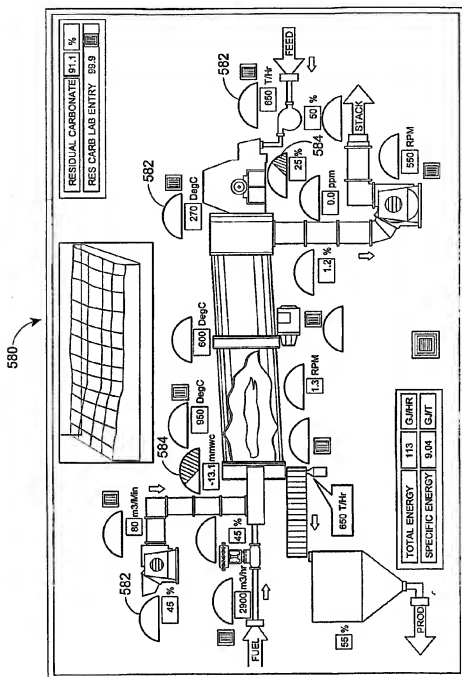


图 20E

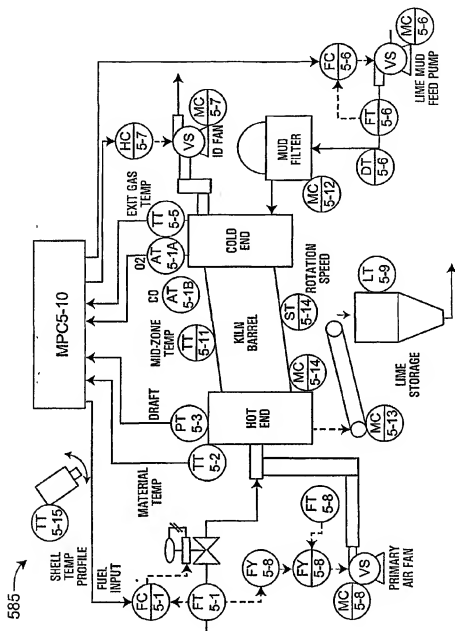


图 21A

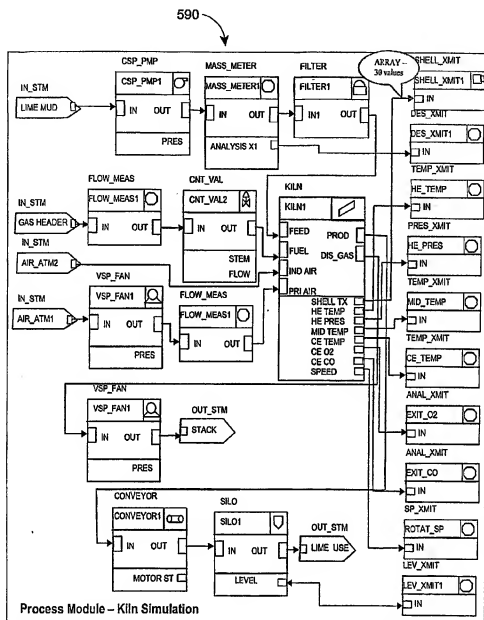


图 21B



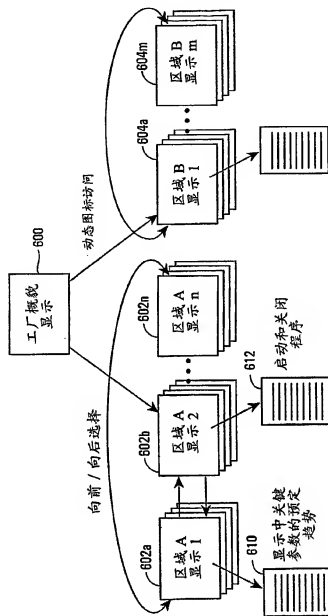


图 22

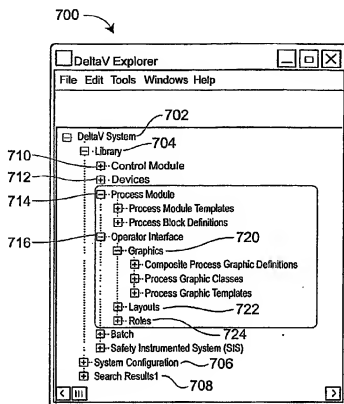


图 23

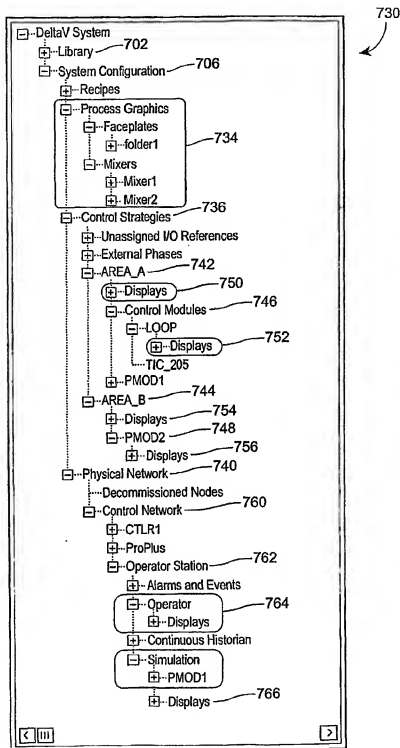


图 24

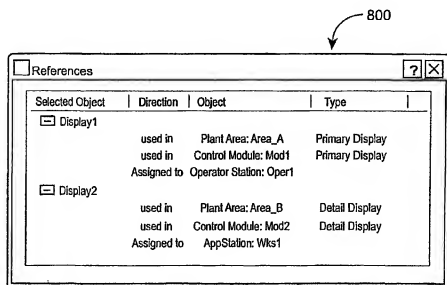


图 25